MASTER EN SCIENCES INFORMATIQUES

La traduction automatique. Etude de cas : Logos

Heymans, Patrick; Rouard, Manuel

Award date:
1995

Facultés Universitaires Notre-Dame de la Paix de Namur
Institut d'informatique

La Traduction Automatique
Etude de cas : Logos

Patrick HEYMANS
Manuel ROUARD

Promoteur : Jacques Berleur s.j.
Co-promoteur : Guy Deville

Mémoire réalisé en vue de l'obtention du grade de Licencié et Maître en Informatique

Année académique 1994-95

RESUME

Le but de ce travail est de présenter le système Logos mis au point par Logos Corporation. Il s'agit d'un système multilingue de Traduction Automatique travaillant sans limitation du domaine discours et sans l'aide d'un opérateur humain lors du processus de traduction. Ce type de système n'est pas en mesure de fournir une traduction satisfaisante, il est donc nécessaire de soumettre le résultat à un traducteur afin qu'il soit post-édité.

Le fonctionnement de ce système étant peu connu actuellement, son explication constitue une part importante de notre travail. Cette explication est précédée d'une introduction générale aux modèles linguistiques et au domaine de la Traduction Automatique afin de donner au lecteur toutes les informations nécessaires à la bonne compréhension des critiques que nous formulons.

La dernière partie de notre mémoire a pour but de situer le système Logos parmi les systèmes les plus réputés dans le domaine de la Traduction Automatique. Les systèmes qui seront développés sont les suivants : Eurotra, Susy, Systran, TAUM-Meteo et Metal.

ABSTRACT

The purpose of this paper is to present the Logos system developped by Logos Corporation. It is a multilingual general purpose Machine Translation system requiring no help from human operator during the translation process. Currently, the output of this type of system is still not satisfactory enough to be used without further refinements. Post-editing by professional translator is consequently still needed.

At present, the way the system works is poorly known. Its explanation represents a major part of our work. This explanation is preceeded by a general introduction to linguistic models and to the field of Machine Translation in order to provide the reader with all the necessary information to make easier the understanding of our critics.

The final part of our thesis is aimed at situating Logos among the main systems in the field of Machine Translation. These systems are : Eurotra, Susy, Systran, TAUM-Meteo and Metal.

REMERCIEMENTS

Nous remercions notre promoteur le Père Jacques Berleur, ainsi que notre co-promoteur Monsieur Guy Deville pour l'aide très précieuse qu'ils nous ont fournie.

Nous tenons également à exprimer toute notre gratitude à l'ensemble du personnel de la compagnie Logos pour l'amabilité et la disponibilité dont ils ont fait preuve tout au long de notre stage. Nous remercions en particulier Monsieur Claus Gehner qui a accepté de nous recevoir ainsi que Kutz Arrieta, Claudia Gdaniec, Liz Purpura, Johnatan Lewis et Bernard Scott qui nous ont consacré beaucoup de leur temps.

Table des Matières

Glossaire

Ce bref glossaire définit les termes linguistiques utilisés dans le mémoire qui peuvent ne pas être connu du lecteur non spécialiste et dont la définition dans le texte aurait nuit à la lisibilité de celui-ci. Notons, d'autre part, que ces définitions n'ont peut-être pas la rigueur requise du point de vue de la linguistique théorique mais elles sont assez précises pour permettre au lecteur de comprendre le présent travail. Les sources que nous avons utilisées pour définir les termes ci-dessous sont [SABAH] et [LAROUSSE].

- **Allotaxie** : Synonymie au niveau de la phrase. On parle donc d'allotaxie quand deux tournures syntaxiques différentes peuvent s'interpréter de la même façon. Le phénomène symétrique est celui des homotaxies.

  Exemple : *Jean casse le vase* et *Le vase est cassé par Jean.*

- **Anaphore** : Références à des objets présentés dans le discours antérieur ou immédiatement postérieur. On distingue plusieurs cas :

  - les pronoms,
  - les références définies,

  Exemple : ***Un** homme se promène. **Ce** garçon est épicier.*

  - l'**ellipse** qui est la forme d'anaphore la plus complexe puisqu'elle n'est marquée que par une absence.

  Exemple : *Paul boit de la bière et Jean - du vin.*

- **Apposition** : Procédé par lequel un terme (nom ou adjectif) ou une proposition qualifient un nom ou un pronom en leur étant juxtaposés [LAROUSSE].

Exemple : *Paris, capitale de la France.*

- **Ellipse** : *voir Anaphore.*

- **Fléchie (forme)** : *voir Racine.*

- **Inflexion (règles d')** : Règles qui, à partir d'une racine d'un mot, décrivent les affixes qu'il faut y ajouter afin d 'en obtenir la forme fléchie désirée (voir aussi *Racine et forme fléchie).*

- **Homographes** : homonymes ayant la même orthographe [LAROUSSE].

- **Homotaxie** : On parle d'homotaxie quand la même tournure syntaxique peut avoir des interprétations différentes. Le phénomène symétrique est celui des allotaxies.

  Exemple : *Prendre le bus pour aller au zoo* et *Prendre son parapluie pour aller au zoo.*

- **Morphème** : Unité minimale de signification. On distingue les morphème grammaticaux comme, par exemple, *-ent* qui marquent la troisième personne du pluriel des verbes, et les morphèmes lexicaux comme, par exemple, *prudent* dans *imprudemment*, *voi-* dans *voient*, etc.

- **Phonème** : Son d'une langue défini par les propriétés distinctives (traits pertinents) qui l'opposent aux autres sons de cette langue [LAROUSSE].

- **Polysème** : Mot qui présente plusieurs sens.

- **Racine et forme fléchie** : D'après [LAROUSSE], la racine est *la forme abstraite obtenue après élimination des affixes et des désinences, et qui est porteuse de la signification du mot*.

  Exemple : *Chanter* a *chant* pour racine.

Les formes fléchies sont les différentes formes d'un mot qu'on peut produire à partir de la (ou des) racine(s).

Exemple : *chant* → *chanter, chanterai, chanterez, etc.*

Introduction

Qu'on le regrette ou qu'on s'en réjouisse, les travaux informatiques sur le langage et la parole sont, comme le constate [BERLEUR], « *[...] l'affleurement "spectaculaire" d'un ensemble de travaux poursuivis depuis près de 40 ans, c'est-à-dire dès la naissance des ordinateurs. Ils constituent aujourd'hui, avec les systèmes experts "la vitrine majeure de l'intelligence artificielle"* ».

En informatique, pour l'ensemble des travaux portant sur le langage naturel, Berleur a relevé cinq directions principales : l'établissement d'outils d'analyse linguistique, l'enseignement des langues assisté par ordinateur, la compréhension du langage naturel, la reconnaissance de la parole et la Traduction Automatique (TA). C'est à ce dernier point que nous avons choisi de nous intéresser même si ce n'est pas, à proprement parler, du point de vue de l'Intelligence Artificielle (IA) que nous allons l'aborder mais en tant qu'une spécialité à part entière à la croisée des chemins de l'informatique et la linguistique[1].

La première partie du travail s'attache précisément à rendre compte de ce qu'est actuellement la TA et de ce qui l'a conduit jusque là. Un premier chapitre est consacré aux notions de linguistiques utiles en TA. Il rappelle la notion importante de niveaux de connaissances linguistiques et illustre leur utilisation par un exemple d'analyse de phrases. La suite du chapitre passe en revue les modèles linguistiques d'intérêt dans le champs d'application étudié. Un second chapitre retrace brièvement l'historique du domaine puis traite des caractéristiques de base des systèmes de TA qui, comme nous allons le voir, sont extrêmement importantes pour la suite du travail.

En effet, dans la seconde partie du mémoire est décrit précisément Logos, un système américain développé par Logos Corporation (Mt. Arlington, New Jersey) où nous avons effectué un stage de quatre mois, de septembre 1994 à janvier 1995. La bonne qualité des résultats rencontrés par le système pousse actuellement la compagnie à faire connaître au public la façon dont fonctionne son système de traduction qui, depuis les débuts de sa conception dans les années 70, n'est resté connu que de manière interne à l'entreprise. Notre

[1] Les liens entre la TA et l'IA seront abordés brièvement dans le travail afin de clarifier un peu cette situation (voir section 4 du chapitre II).

tâche principale pendant le stage a donc été de nous familiariser avec ce système particulièrement complexe, tâche qui a été rendue difficile par le manque de publications détaillées sur le sujet. Nous dûmes donc nous orienter vers deux méthodes de travail : la confrontation directe avec le système par observation des *diagnostics* (traces) que fournit celui-ci lors des traductions et la discussion avec des membres du personnel afin d'interpréter les résultats et d'acquérir ainsi une plus grande compréhension du fonctionnement du système. Un second apport dont nous avons fait bénéficier Logos est la participation au *Translatability Index Project*, un projet expérimental testant la faisabilité d'un outil d'estimation a priori de la qualité des phrases à traduire. Le chapitre III décrit le système Logos de façon détaillée en y incluant l'explication du *Translatability Index Project*. Dans le chapitre IV, nous analysons le système de manière critique en en pesant le pour et le contre.

Enfin, dans la dernière partie du travail, nous poursuivons un double objectif. Il s'agit, d'une part, d'étendre notre vision du domaine de la TA en consacrant un chapitre à chacun de cinq des systèmes parmi les plus marquants : Eurotra, Susy, Systran, TAUM-Meteo et Metal. D'autre part, cela nous permet d'affiner notre critique du système Logos. En effet, chaque chapitre consacré à un système donné se termine par une critique dans laquelle, lorsque cela nous a semblé pertinent, nous avons relevé des points de comparaisons avec Logos. Tous ces systèmes travaillant sur des langues différentes et n'ayant pas forcément la même utilisation, la comparaison ne pourra se faire sur la qualité des traductions. Celle-ci se fera donc principalement sur les caractéristiques de base des systèmes vues au chapitre II.

Nous concluons en effectuant un bref parallèle entre tous les systèmes développés dans la troisième partie et le système Logos.

Première partie :

Notions de linguistique et de Traduction Automatique

Chapitre I : Notions de linguistique

1. *INTRODUCTION*

Nous l'avons vu dans l'introduction générale, la linguistique est une science dont le champ d'investigation est très vaste et qui collabore avec d'autres disciplines. Dans ce chapitre, nous allons nous limiter à donner les notions de linguistique qui sont utiles en informatique pour le traitement du langage naturel et, plus particulièrement, pour la TA.

[SABAH] part du principe que : « *Pour toutes les applications informatiques utilisant les langues, les mêmes processus de base sont nécessaires : il faut "comprendre effectivement" un texte pour effectuer une tâche quelconque sur ce texte, on doit ensuite utiliser des processus de "raisonnements intelligents" puis traduire les éléments de réponse trouvés en langue naturelle. Enfin, des procédures de gestion du dialogue peuvent être nécessaires dans certaines applications* ».

[SABAH] décrit en fait ici un seul type de système — nous en verrons d'autres au point 2.3 du chapitre II — mais qui est de loin le plus complet au niveau des connaissances qu'il met en oeuvre, les autres n'utilisant que divers sous-ensembles de ces connaissances. Pour lui, le fonctionnement du système se divise classiquement en trois phases :

- **l'analyse** : construction d'une "représentation interne", généralement formée d'une association de symboles sensés traduire le sens[2] de la phrase et du texte,
- **les inférences** : raisonnements effectués par le système pour calculer la réaction adéquate à ce qui a été dit (représentation interne),
- **la génération** : transmission des conclusions des raisonnements du système à l'utilisateur. Dans notre cas, il s'agit de produire le texte traduit à partir de la représentation interne et des inférences qu'elle a suscitées.

[2] Nous verrons au point 3.3 du chapitre II que la représentation interne de la plupart des systèmes se limite souvent à traduire la structure du texte ou de la phrase et ne traduit son sens que dans une moindre mesure.

Venant ainsi de citer les opérations que sont sensés pouvoir réaliser les systèmes de TA, nous allons, à la section 2 de ce chapitre, décrire les différents niveaux de connaissances linguistiques requis pour y parvenir. A la section 3, nous décrirons les principaux modèles linguistiques qui utilisent ces connaissances et qui sont utiles en TA.

2. *LES NIVEAUX DE CONNAISSANCES LINGUISTIQUES*

2.1. Connaissances morphologiques et lexicales

Par **connaissances lexicales**, on entend l'ensemble des informations attachées à chaque élément lexical (mot ou groupe de mots) qui fait partie du vocabulaire d'une langue donnée. Les éléments lexicaux sont répartis en **catégories lexicales** telles que : nom, adjectif, pronom, etc. Les connaissances lexicales d'un système sont rassemblées dans des **lexiques** qui sont aussi parfois appelés dictionnaires mais dont le contenu diffère assez fortement de ce que l'on retrouve dans ce que l'on appelle communément dictionnaire. Une entrée de lexique (qui peut être une racine ou une forme fléchie complète) a pour but de donner de l'information pour les traitements syntaxiques et sémantiques. C'est pourquoi on y retrouve des renseignements tels que la catégorie syntaxique (voir point 2.2 ci-dessous), des informations morphologiques (voir ci-dessous), les valences (voir point 3.4 de ce chapitre), les "case frames" (voir point 3.5 de ce chapitre), des informations sémantiques, des restrictions quant à leur sélection, etc.

Dans les phrases soumises au système, un mot peut apparaître sous des formes diverses (par exemple *viens, viendra, venir*). Une approche qui est souvent considérée comme naïve est d'associer à chaque forme une entrée du dictionnaire du système et de répéter plusieurs fois les mêmes informations (par exemple, "verbe intransitif"). La solution qui est la plus souvent adoptée consiste à reconstruire la forme de référence à chaque occurrence du mot. Dans une solution comme dans l'autre, on utilise des **connaissances morphologiques** qui sont :

- la forme **canonique** d'un mot c'est-à-dire l'infinitif pour un verbe, les masculin singulier pour un nom ou un adjectif, etc.
- ses racines et ses terminaisons possibles,
- les règles qui régissent les associations des racines et des terminaisons.

2.2. Connaissances syntaxiques

« Ce sont des connaissances générales qui expriment des relations formelles entre des catégories de mots, indépendamment de leur contenu sémantique et des sujets qui les utilisent. Elles sont également désignées par le terme de grammaire » [SABAH].

Il s'agit en fait des connaissances qui, une fois reconnues les formes canoniques des mots de la phrase, permettent de voir comment ceux-ci sont agencés : quels sont les compléments de la phrase?, etc. Les différents constituants du textes peuvent être répartis en **catégories syntaxiques** telles que : sujet, verbe, objet direct, etc. Mais une phrase syntaxiquement correcte n'est pas forcément une phrase compréhensible. D'autres niveaux de connaissances sont nécessaires pour lever les ambiguïtés restantes et tenter de "comprendre" le texte.

2.3. Connaissances sémantiques

« On considère couramment qu'elles représentent le "sens" des mots. Plus concrètement, on peut dire (avec une conception référentielle du sens) qu'elles expriment les relations qui existent entre les mots du langage et les objets ou les actions du monde (le monde réel ou le monde particulier dont on parle), sans référence au locuteur. Elles correspondent donc aux aspects explicites du sens » [SABAH].

2.4. Connaissances pragmatiques

Ce sont les connaissances qui rendent compte des aspects *implicites* du sens. Elles permettent de réduire certaines ambiguïtés en permettant la bonne interprétation du texte en fonction :

- du contexte,
- de connaissances générales.

Parmi ces deux formes d'interprétation, la première représente les aspects conventionnels des langues et traduit des relations qui existent entre les mots du langage et celui qui les interprète. La seconde, plus couramment utilisée en intelligence artificielle, est une interprétation de l'énoncé par rapport aux objets désignés. Il s'agit en fait d'un ensemble de connaissances sur le monde de référence et qui constituent la culture des interlocuteurs et qui, tout en restant implicites, permettent aux interlocuteurs de se comprendre.

2.5. Un exemple

Voyons comment sont appliqués à l'analyse de la phrase *le secrétaire vole des livre* les différents types de connaissances développés ci-dessus. Cette phrase comporte les ambiguïtés suivantes :

- *le* = article ou pronom
- *secrétaire* = homme, meuble ou oiseau
- *vole* = planer ou dérober
- *des* = article indéfini, article contracté ou article partitif
- *livres* = bouquin, monnaie ou poids

A priori, il y a 108 façons d'analyser la phrase. Mais on va voir comment l'ajout de connaissances linguistiques à cette analyse permet d'en réduire le nombre.

L'analyse morphologique de la phrase donne pour chaque mot, l'ensemble des catégories lexicales et des formes qui peuvent lui correspondre (le nombre de catégorie lexicales est mentionné entre parenthèses).

- *le* = article masculin singulier (2)
- *secrétaire* = nom masculin singulier (1)
- *vole* = verbe voler (transitif) 1ère/3ème personne indicatif présent etc. ou verbe voler (intransitif) 1ère/3ème personne indicatif présent etc. (2)
- *des* = article indéfini, article contracté ou articles partitif pluriel (3)
- *livres* = nom masculin pluriel ou nom féminin pluriel (2)

Il reste à ce niveau 24 (2*2*3*2) combinaisons possibles.

Au niveau syntaxique, les règles de grammaire nous apprennent que :

- *le* est un article
- *vole* est un verbe transitif (avec les sens de dérober)

Par contre, les ambiguïtés morphologiques sur *des livres* restent entières tout comme le fait de savoir si *le secrétaire* est un homme, un meuble ou un oiseau.
Au niveau sémantique, on apprend que *voler* (transitif) n'admet qu'un agent animé comme sujet, ce qui élimine l'interprétation de *secrétaire* en tant que meuble. D'autre part, *voler*

(transitif) requiert un objet concret, ce qui élimine l'interprétation de *livres* comme "mesure de poids".

La levée de l'ambiguïté liée à l'article *des* est un peu plus délicate car elle se situe à la limite de différents types de connaissances. En effet, ce sont les caractéristiques syntaxiques de *voler* (transitif) et les caractéristiques sémantiques de *livre* qui permettent de résoudre cette question.

A ce niveau de l'analyse, il ne reste plus que quatre interprétations qui sont représentées à la figure 1.1 ci-dessous :

Figure 1.1 : Quatre interprétations de *le secrétaire vole des livres* (emprunté à [SABAH])

La "désambiguätion" de la phrase sera terminée par l'interprétation qu'on peut faire en fonction de la situation de la communication et de notre représentation du monde (connaissances pragmatiques).

2.6. Remarques importantes

Nous attirons d'abord l'attention sur le fait que, si l'on se situe dans une perceptive scientifique, la séparation des connaissances linguistiques en niveaux bien distincts ne se justifie que comme support à l'étude de phénomènes de base. L'exemple du point 2.5 montre bien que, pour "comprendre" une phrase, tant l'homme que la machine sont amenés à travailler à différents niveaux simultanément (notamment syntaxique et sémantique) et à résoudre des problèmes plus ou moins imbriqués (dans l'exemple, la résolution de l'ambiguïté de *livres* rend accès aux traits sémantiques qui peuvent permettre la résolution des ambiguïtés de *des*).

Cette remarque est extrêmement importante car, comme on le verra par la suite, beaucoup de systèmes (se basant sur des critères de modularité) ont mis en oeuvre des approches linguistiques qui empêchent la collaboration de différents niveaux de connaissances linguistiques, un module s'occupant uniquement de traitements à un niveau de connaissances linguistiques déterminé.

Une deuxième remarque concerne l'ordre dans lequel nous avons vu les niveaux de connaissances linguistiques. L'ordre suivi est celui de l'analyse mais il est clair que ces connaissances sont également utilisées pour la génération, mais cette fois dans l'ordre inverse.

3. LES MODELES LINGUISTIQUES

Les connaissances linguistiques dont nous avons parlé à la section précédente ne peuvent être utilisées dans des systèmes informatiques que si l'on dispose de formalismes pour les représenter. Certains modèles linguistiques nous fournissent de tels formalismes. Et ils permettent aussi de représenter les phrases en cours de traitement.

Ces modèles étant essentiellement basés sur la formalisation des connaissances syntaxiques, on les appelle communément **grammaires**. La notion de grammaire est définie par [SABAH] comme *« un ensemble de règles qui régissent les possibilités d'association des mots entre eux selon leurs catégories lexicales et qui permettent de décider si une phrase donnée est correcte ou non »*.

3.1. Les grammaires formelles

3.1.1. Notions de base

Le *vocabulaire* (V) du langage est décomposé en deux sous-ensembles finis disjoints : le *vocabulaire terminal* (VT, c'est-à-dire l'ensemble des symboles qui peuvent apparaître dans les phrases de la langue) et le *vocabulaire non terminal* (VN, c'est-à-dire l'ensemble des symboles nécessaires à la description de cette langue et qui sont en fait les catégories syntaxiques).
Un langage sur VT est alors défini comme un ensemble (potentiellement infini) de chaînes de longueur finie formées avec des éléments de VT.

Les *règles de réécriture* (R, encore appelées *règles de production*) spécifient les relations permises entre chaînes formées de symboles de V et permettent ainsi de rendre compte de la structure interne de la phrase.

Exemples

(1) P → GN + GV

(2) NOM → *garçon*

Si l'on décide d'interpréter "+" comme un symbole de concaténation et la flèche comme une instruction ordonnant de réécrire le symbole de gauche en utilisant les symboles de droite, (1) se lit comme la règle grammaticale qui dit : "une phrase peut être composée d'un groupe nominal (GN) suivi d'un groupe verbal (GV)".

Dans les langues naturelles, on distingue deux types de règles. Les premières, comme (1), explicitent des relations entre catégories syntaxiques, les autres, appelées règles de "lexicalisation" transforment une catégorie lexicale en un mot du lexique (symbole terminal). (2) est un exemple de ce dernier type de règle.

Enfin, parmi les symboles non terminaux, on distingue un symbole particulier : le *symbole origine* (noté P pour phrase).

Une grammaire formelle est alors définie par le quadruplet : G=(VN,VT,R,P) et l'on appelle *langage engendré par la grammaire G*, l'ensemble de toutes les chaînes terminales que l'on peut dériver du symbole d'origine en appliquant toutes les séquences possibles de règles de réécriture.

Les grammaires formelles sont regroupées en quatre types numérotés de 0 à 3 par Chomsky. Nous les passons en revue ci-dessous, partant des plus simples, en montrant, les insuffisances et aboutissant ainsi aux plus générales.

3.1.2. Grammaires de type 3

Ces sont des grammaires dites régulières. Si X et Y sont des éléments non terminaux et si a est un élément terminal (X,Y ∈ VN et a ∈ VT), les règles sont de la formes :

- X → a + Y ou X → a pour les grammaires régulières à gauche,
- X → Y + a ou X → a pour les grammaires régulières à droite.

Dans ces grammaires, la connaissance de tout ce qui précède (ou suit, selon la forme des règles) est inutile pour trouver la continuation correcte de la phrase. Ce type de grammaire est donc insuffisant pour traiter les divers aspects de la langue et, en particulier, les structures imbriquées ou parenthèsées. Cependant, comme elles peuvent générer un langage fini

(composé d'un nombre fini de phrases), les grammaires régulières peuvent être utilisées pour des applications fortement simplifiées. On peut montrer que ce formalisme correspond à la notion d'automate fini.

3.1.2. Grammaires de type 2

Dans les règles de ces grammaires, le membre de gauche ne peut être qu'un (et un seul) symbole non terminal. Elles sont dites *indépendantes du contexte* ou *non contextuelles* car les règles signifient que le membre de gauche peut être réécrit sous la forme de droite, indépendamment du contexte (les symboles qui l'entourent). Ci-dessous nous trouvons un exemple simplifié d'un fragment de grammaire de la langue française.

- (R1) P → GN + GV
- (R2) GN → ART + NOM
- (R3) GV → (AUX)+VERBE+GN
- (R4) AUX → va
- (R5) VERBE → lire / bat / mange / ...
- (R6) ART → le / la / les / un / une / ...
- (R7) NOM → garçon / livres / pomme / ...

Cette grammaire permet notamment de déduire et de produire la phrase *le garçon va lire des livres* par applications successives des règles (R1), (R2), (R3), (R2), (R6), (R7), (R4), (R5), (R6), (R7).

Un des avantages de ce type de grammaire est de permettre une représentation des applications de règles aux phrases sous la forme d'*arbres de dérivation* communément appelés *arbres de dépendances* (voir figure 1.2 ci-dessous).

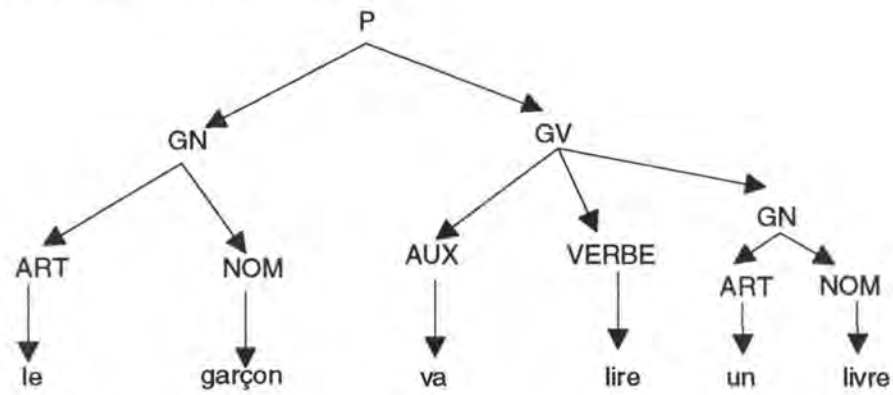


Figure 1.2 : Arbre de dérivation/dépendances de la phrase *le garçon va lire un livre* (emprunté à [SABAH])

Cependant, comme le laisse sous-entendre les représentations graphiques de ces types de grammaire, il leur est impossible de traiter les constituants discontinus (par exemple *ni ... ni*). Ce formalisme correspond à la notion d'automate en pile et est donc aisément implémentable pour des applications très restreintes. Mais bien qu'on ait jamais pu le prouver, Chomsky a émis l'hypothèse que la syntaxe des langues n'était pas formalisable par des grammaires non contextuelles. Même en faisant abstraction de la remarque de Chomsky, on constate dans la pratique que le nombre de règles qu'il faudrait pour rendre compte de tous les phénomènes d'une langue est tellement grand qu'un traitement informatique deviendrait impossible.

3.1.3. Grammaires de type 1

Les règles sont ici de la forme :
$u + x + v \rightarrow u + y + v$
(avec $u,v,y \in V^*$ c'est-à-dire des chaînes quelconques formées par combinaisons d'éléments du vocabulaire, $x \in VN$ et $y \neq \varnothing$ la chaîne vide)

Ces grammaires sont dites *sensibles au contexte* ou *contextuelles*. On comprend mieux cette application en paraphrasant la règle-type ci-dessus en : "*x peut se réécrire en y dans le contexte u ... v*".

On peut montrer que les langages engendrés par les grammaires de ce type peuvent être reconnus par des automates linéaires bornés (un processus informatique déterministe utilisant une mémoire proportionnelle à la longueur de la phrase entrée).

La puissance de ces grammaires peut être augmentée par l'adjonction de *traits* qui ne correspondent pas forcément à des mots. Il s'agit en particulier, comme nous le montre l'exemple suivant, de la notion de genre et de nombre.

- NOM + Pluriel → NOM + *s*
- VERBE + Pluriel → VERBE + *ent*

Dans l'exemple suivant, on montre qu'on peut tenir compte de l'accord sujet verbe :

- P → GN + GV
- GN → ART + NOM + Nombre
- Nombre → Singulier
- Nombre → Pluriel

- Singulier + GV → Singulier + VERBE + (GN)
- Pluriel + GV → Pluriel + VERBE + (GN)

En fait, cela peut être fait dans une grammaire contextuelle mais cela requiert une duplication de l'ensemble des règles où le *trait* intervient :

- P → P-sing
- P → P-plur
- P-sing → GN-sing + GV-sing
- P-Plur → GN-plur + GV-plur

Certains phénomènes des langues (accord article-nom-adjectif ou sujet-verbe, distributivité, etc.) ont permis de souligner qu'il ne s'agissait ni d'un langage régulier, ni d'un langage non contextuel. Ces grammaires contextuelles furent elles aussi rejetées, sans véritable preuve mais aux vues des constatations suivantes :

- maladresse et complexité des descriptions de règles
- difficulté de traiter des allotaxies, homotaxies, constituants discontinus.

Ces grammaires contextuelles étant de plus relativement difficiles à mettre en oeuvre sur le plan informatique, la plupart des systèmes qui ont fait utilisation de grammaires formelles ont utilisé des grammaires non contextuelles dont on a tenté d'augmenter la puissance expressive de manières diverses pour les rendre équivalentes aux grammaires contextuelles.

3.1.4. Grammaires de type 0

Aucune contrainte n'est introduite sur les règles de réécriture. Les langages correspondants sont reconnus par une machine de Turing générale. Mais un tel système est trop peu structuré et ne peut en pratique servir de grammaire.

3.2. Grammaires transformationnelles

Dans la version standard de cette théorie, due à Chomsky, on a une organisation des connaissances syntaxiques en trois volets :

- une grammaire formelle permettant d'engendrer un ensemble de structures abstraites dites *structures profondes*. Il s'agit en fait de phrases noyaux (simples, déclaratives, à la forme active) auxquelles on associe un arbre de dérivation.
- un ensemble de règles de transformation qui agissent sur les arbres des phrases noyaux en réordonnant les chaînes terminales, en y ajoutant ou en supprimant des éléments. Les structures ainsi créées sont appelées *structures de surface* et on engendre ainsi toutes les formes possibles des phrases.
- des règles morpho-phonémiques permettant de construire à partir des chaînes terminales la suite de phonèmes.

La théorie peut être schématisée par la figure 1.3 ci-dessous :

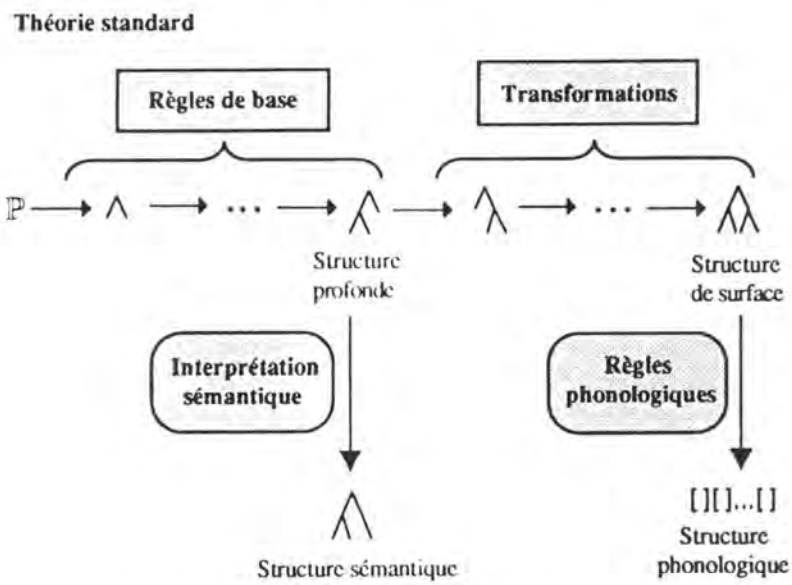


Figure 1.3. : Un modèle de la *théorie standard* (emprunté à [SABAH])

La structure profonde est interprétée par un composant sémantique qui produit le sens de la phrase engendrée. A partir de la structure de surface est engendrée la suite de sons ou de mots qui constitue la phrase. Ce modèle implique une autonomie de la syntaxe qui reste très discutable en linguistique.

Un aspect important de cette théorie est la distinction qui est faite entre structure profonde et structure de surface. Aux figures 1.4 et 1.5 ci-dessous, nous illustrons le fait qu'une même structure profonde peut donner lieu (par l'intermédiaire de transformations différentes) à plusieurs structures de surface et le fait inverse, à savoir que deux structures profondes distinctes peuvent se réaliser en la même structure de surface.

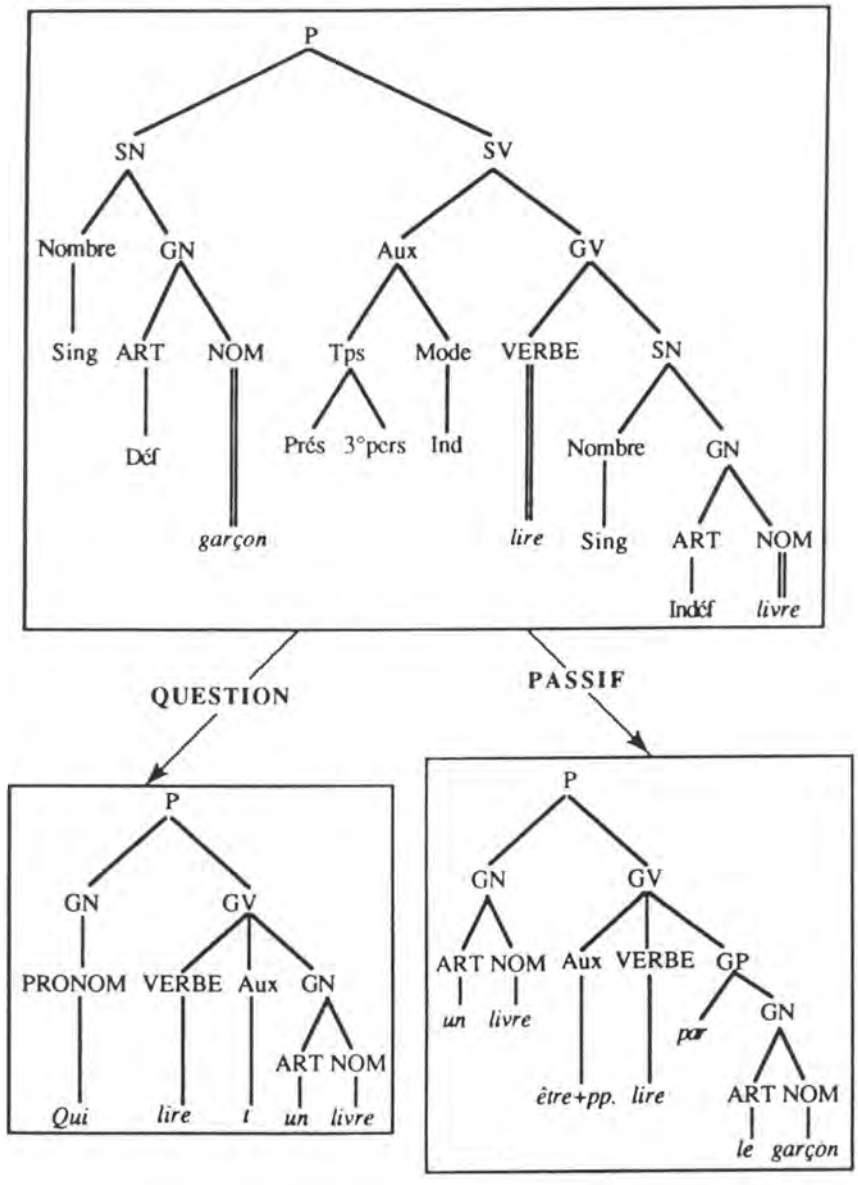


Figure 1.4 : Structures de surfaces issues d'une même structure profonde (emprunté à [SABAH])

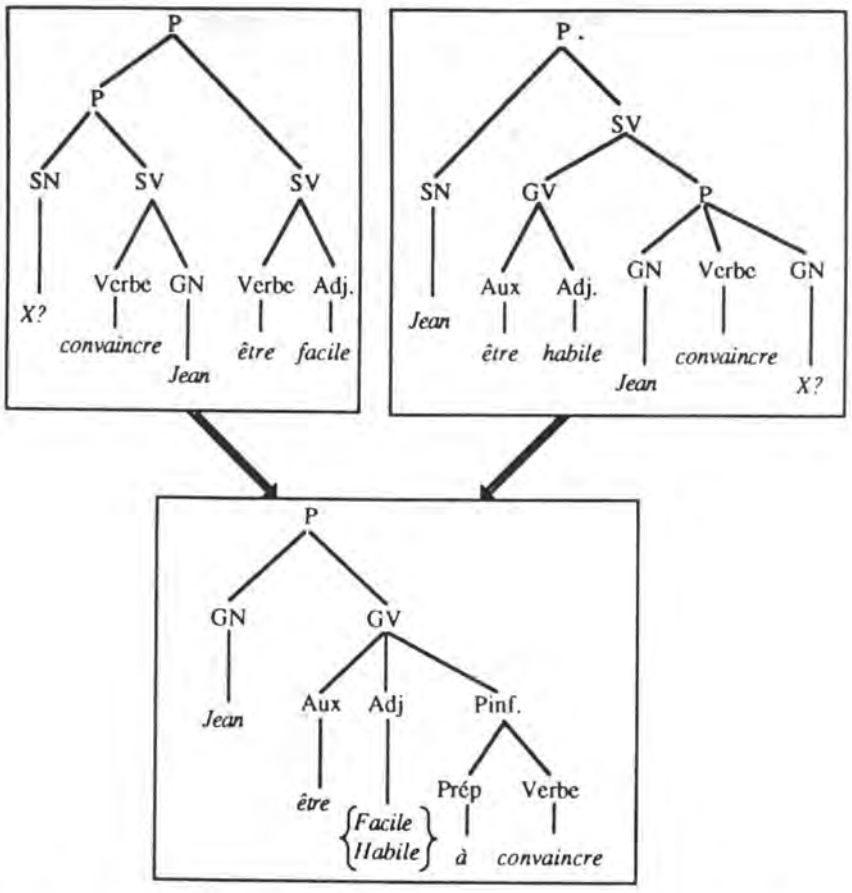


Figure 1.5 : Structures profondes distinctes produisant la même structure de surface (emprunté à [SABAH])

Une *transformation* est constituée :

- d'une *description structurelle* (DS) donnant la forme de la partie de l'arbre à laquelle on applique la transformation.
- de *changements structurels* (CS) qui sont des opérations à effectuer sur l'arbre quand la règle est appliquée.
- des *conditions* précisant les cas où la règle s'applique.

Généralement DS spécifie une séquence de noeuds voisins sans préciser leurs relations exactes dans l'arbre. Les CS sont conçus comme une série de *transformations élémentaires* qui sont soit des suppressions (symbolisée par 0), des *substitutions* ou des *adjonctions*. On distingue les adjonctions comme frère (+), fils le plus à droite (>), fils le plus à gauche (<) et les adjonctions "complexes" (#).

A la figure 1.6 ci-dessous, on trouve l'exemple de la transformation passive :

PASSIF (Optionnel)

DS :	GN	Aux	VERBE	GN
	1	2	3	4
CS :	4	2 > être + pp.	3	par # 1

Cette transformation permet le passage (facultatif) :

Paul lit des livres ⟶ *Des livres sont lus par Paul.*

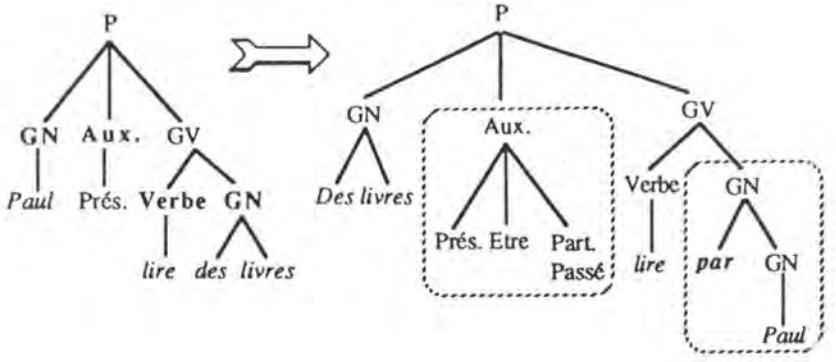


Figure 1.6. : Application d'une transformation (emprunté à [SABAH]).

Les transformations peuvent être *facultatives*, comme pour les changements de voix (voir figure 1.6 ci-dessus) ou *obligatoires* comme quand il y a des traits dans la structure profonde ou, comme le montre la figure 1.7, pour supprimer des GN égaux.

SUPPRESSION-GN-EGAUX (obligatoire)

DS	:	GN	Verbe	que	GN	AUX	VERBE
		1	2	3	4	5	6
CS	:	1	2	0	0	5+inf	6
Conditions :		1) 1=4					
		2) type de 2 = volonté, croyance, ...					

Figure 1.7. : Transformation obligatoire de GN égaux (emprunté à [SABAH])

Une remarque importante est que les transformations ne doivent pas changer le sens puisque l'interprétation sémantique se situe au niveau de la structure profonde. Il est donc nécessaire que certains traits comme la négation apparaissent dans la structure profonde. D'autre part, l'ordre dans lequel s'effectuent les transformations a son importance puisque, par l'effet de substitutions, certains éléments nécessaires à une transformation peuvent lui avoir été rendus indisponibles par une transformation précédente.

La théorie des grammaires transformationnelles souffre cependant de deux défauts majeurs :

- sa puissance d'expression trop importante (démontrée équivalente à une grammaire de type 0) fait qu'en pratique, il peut s'avérer presque impossible pour un analyseur de dire quelle règle de transformation a été employée.
- elle considère que le sens est déterminé par la structure profonde, **avant** l'application de transformations. Si on prend l'exemple de la phrase *Peu de gens lisent beaucoup de livres* et sa paraphrase *La plupart des gens lisent peu*. La transformée passive de la première est *Beaucoup de livres sont lu par peu de gens* qui peut être paraphrasée en *Il y a beaucoup de livres impopulaires*, ce qui diffère fortement de la première paraphrase au niveau du sens. En général, le peu de prise en compte de la sémantique dans les grammaires transformationnelles fait qu'on considère qu'elles ne sont pas suffisantes pour qu'il y ait "compréhension" du texte.

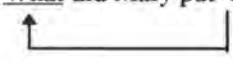
Le formalisme des grammaires transformationnelles est à l'heure actuelle dépassé. Cependant, on en retrouve encore l'emprunt dans de nombreux systèmes de TA existants, surtout au niveau de la distinction qu'il fait entre structure profonde et structure de surface.

3.3. La théorie des traces

La théorie des *traces* est une extension de la théorie standard des grammaires transformationnelles qui entend rendre compte des mouvements implicites des divers constituants de la phrase, en particulier dans le cas d'infinitives et relatives. Elle rejette plus ou moins l'idée de structure profonde en supposant que le composant sémantique et le composant phonologique agissent tous deux sur une structure de surface où sont indiquées les positions (traces, notées τ) des groupes déplacés par le composant transformationnel.

Exemple

« What did Mary put τ on the table »



Une généralisation de la théorie des traces (appelée grammaire syntagmatique généralisée) permet également de traiter les éléments qui ont été simplement effacés (les ellipses). Ces éléments sont appelés "trous" (noté χ)

Exemple

« John likes fish, and Bill χ meat »

Il était important de soulever la problématique des traces et des "trous" car ceux-ci sont souvent difficilement traités par les systèmes de traduction.

3.4. Grammaires de valences

Le principe de la grammaire des valences est d'associer à chaque verbe un pattern qui donnent le nombre et la nature des arguments (appelés parfois *cas syntaxique*) ainsi que la forme syntaxique que prend la construction. Généralement la *valence* d'une verbe (nombre de ses arguments) s'échelonne de 1 à 3. Il peut s'agir d'un sujet (obligatoire sauf pour les verbes impersonnels comme *pleuvoir*), d'un objet (indirect, noté OD, ou indirect, noté OI) ou d'un complément prépositionnel). Les structures syntaxiques attachées au verbe sont données par des descriptions comme "fournir *quelque chose* à *quelqu'un*". En mettant l'accent sur les propositions employées. Ces descriptions peuvent même être enrichies de restrictions sémantiques sur les arguments. Dans l'exemple ci-dessus, on pourrait stipuler que le sujet et l'objet indirect de *fournir* doivent être humains et que l'objet direct doit être un produit.

Les grammaires des valences ont deux utilités principales pour la traduction et pour la Traduction Automatique :

- bien que les verbes de différentes langues ne partagent pas les mêmes structures syntaxiques, leurs valences sont souvent identiques et des correspondances peuvent donc être identifiées,
- elles permettent de savoir comment traduire une préposition en connaissant le verbe et le rôle syntaxique de l'argument. Par exemple, à "fournir quelque chose (OD) à quelqu'un (OI)", on fait correspondre "to supply someone (OI) with something (OD)" et on voit qu'en anglais l'objet direct de *supply* est introduit par *with*.

Notons enfin que l'on peut également associer une structure de valence aux adjectifs (par exemple *fier de*) et aux noms (par exemple *la lutte contre ...*).

3.5. Grammaires de cas

Cette théorie, due à Fillmore, soutient que l'on peut identifier un ensemble de cas sémantiques permettant de mettre en évidence, à la manière des *cas syntaxiques* (sujet, verbe, objet, ...) les relations qui existent entre les noms (ou les groupes nominaux) et le verbe de la phrase principale.

La nécessité d'une telle approche se justifie par le fait que trois phrases comme *Jean casse la branche avec une pierre*, *La pierre casse la branche* et *La branche casse* qui décrivent la même action, ont chacune un sujet (fonction de surface) différent, respectivement *Jean*, la *pierre* et la *branche*.

Pour Fillmore, la représentation profonde d'une phrase (plus axée sur la sémantique que la structure profonde de Chomsky) est composée :

- d'une *modalité* : informations sur la négation, le temps, le mode et l'aspect,
- d'une *proposition* qui reconnaît les relations sémantiques (cas) qui lient les groupes nominaux au verbe (composante centrale de la phrase).

Ce qui est intéressant c'est que le nombre de cas est relativement limité (une dizaine, en général) et que les cas sont des relations sémantiques communes à toutes les langues, ce qui est extrêmement intéressant dans le cadre de la TA. Un cas n'ayant (au maximum) qu'une seule réalisation pour un verbe donné dans une phrase donnée[3] et ayant constaté qu'il était possible d'attacher à chaque verbe ses cas sémantiques (obligatoires ou facultatif), il est possible d'attacher à un verbe une *ossature* (on emploie plus couramment le terme anglais "case frame") qui en définit la forme canonique. Pour *casser*, on aurait par exemple : [Objet,(Instrument),(Agent)] (1).

L'intérêt des grammaires de cas pour le traitement automatique des langues est très important et les emprunts à ce type de grammaire sont très nombreux en TA. Les principaux problèmes qui peuvent ainsi être résolus sont :

- le choix du sujet : en ordonnant les cas dans l'ossature (1), on a [(Agent),(Instrument),Objet] qui nous apprend qu'à la forme active, l'instrument peut être sujet, l'agent étant absent, et que l'objet ne peut être sujet que si l'agent et l'instrument sont absents.

[3] A ce niveau, on ne tient compte que de phrase simple sans phénomènes complexes tels que la coordination.

- la détermination du rôle d'un constituant : l'adjonction de restrictions sémantiques, de la même manière que pour les cas syntaxiques, permet de déterminer quel est le cas sémantique d'un constituant.
- les problèmes d'homographes : un verbe peut se voir associer différentes ossatures qui correspondent à ses différents sens.

Exemple

voler [Agent], voler [Objet], voler [Agent, Objet], distinguent le vol d'un oiseau (Agent), du vol d'un avion (Objet) et de l'action de dérober.

Cependant, malgré le grand intérêt que présentent les grammaires de cas, il convient de mettre en garde contre deux de leurs faiblesses. Premièrement, l'ossature attachée à un verbe entraîne que seuls certains aspects de la situation décrite vont être effectivement considérés et compris. Par exemple, si on a convenu que l'ossature de *jouer* (dans le sens jouer de la musique) est [Agent,(Instrument),Temps], dans la phrase *Roger a joué du jazz avec sa guitare toute la nuit pour Jeanne*, on peut se demander quels sont les rôles sémantiques de *jazz* et *Jeanne*. Il y a donc une part d'arbitraire dans les cas que l'on décide de traiter. Ensuite, bien qu'on ait tenté de décrire par des règles comment décider du rôle d'un éléments dans la phrase, ces règles souffrent de nombreux contre-exemples et se prêtent donc difficilement à un traitement automatique efficace.

Exemple

Pour l'anglais, on pourrait trouver :

*« L'agent est introduit par **by** au passif; s'il n'y a pas d'agent, l'instrument est introduit par **by** sinon par **with** etc... »*

Cependant, d'une part, il faut souligner que des propositions ont été faites pour tenter de définir plus formellement la notion de cas, par exemple dans le domaine de la reconnaissance de la parole [DEVILLE] et que, d'autre part, constituant un modèle simple de la sémantique des phrases, les grammaires casuelles sont fortement utilisées en TA.

D'autre part, les grammaires de cas permettent une analyse sémantique fondée sur les restrictions de sélection issues des ossatures des verbes et n'excluant pas l'utilisation de contraintes syntaxiques, même si la "compréhension" du texte qui en résulte reste limitée.

3.6. Grammaires d'unification

Les grammaires d'unification font partie des théories grammaticales qui donnent au lexique une importance primordiale. Elles développent des analyses fondées principalement sur les caractéristiques syntaxiques et sémantiques des **mots** de la phrase. Ces grammaires, comme les grammaires lexicales fonctionnelles que nous verrons juste après (voir point 3.7 ci-après) tentent de résoudre les problèmes des grammaires formelles et transformationnelles grâce à la notion de *description additive* permettant des descriptions partielles qui peuvent se compléter par la suite.

Le fondement de ces théories consiste à considérer de façon unifiée les connaissances lexicales et les connaissances grammaticales comme des expressions de contraintes. De plus les éléments du dictionnaire, les règles de grammaire et les structures internes des phrases sont représentés suivant le même formalisme : le formalisme des *descriptions fonctionnelles* (DF).
Un constituant (*feature* en anglais) est décrit par un ensemble de couples (*feature bundles* en anglais) {(attribut=valeur)}.

Un *schéma* est représenté dans une "boîte" symbolisée par des crochets "[ ]". L'empilement des boîtes les unes dans les autres traduit une structure. Si à un niveau donné plusieurs solutions sont possibles, on représentera cela par plusieurs boîtes réunies par une accolade.
Les attributs les plus fréquents sont :

- CATEGORIE : attribut obligatoire spécifiant la catégorie syntaxique de base du constituant.
- LEX : donne la réalisation lexicale du constituant, c'est-à-dire le mot lui-même.
- FORME : indique les contraintes sur l'ordre des divers constituant impliqués.

On fait aussi fréquemment utilisation d'attributs représentant des traits sémantiques d'entrées lexicales comme (sem=agent animé, etc.).
La figure 1.8 ci-dessous donne une représentation de l'entrée lexicale ambiguë de *la* (article, nom ou pronom).

$$\left\{\begin{array}{l}\left[\begin{array}{l}Catégorie = Article\\Type = Défini\\Nombre = Singulier\\Genre = Féminin\\Lex = La\end{array}\right]\\\left[\begin{array}{l}Catégorie = Pronom\\Type = Personnel\\Nombre = Singulier\\Personne = Troisième\\Genre = Féminin\\Lex = la\end{array}\right]\\\left[\begin{array}{l}Catégorie = Nom\\Nombre = singulier\\Genre = Masculin\\Lex = la\end{array}\right]\end{array}\right.$$

Figure 1.8 : Les trois interprétations du mot *la* (emprunté à [SABAH])

La figure 1.9 ci-dessous donne un exemple d'utilisation du formalisme pour une grammaire n'autorisant que les phrases de type GN+GV+GN ou GN+GN+GV (le symbole "*" indique que l'élément qui suit est facultatif ou répété un nombre quelconque de fois).

$$\left\{\left[\begin{array}{l}Catégorie = Phrase\\Forme = (GN\ GV\ GN)\\GN = \left\{\begin{array}{l}\left[\begin{array}{l}Cat = GN\\Forme = (DET\ *ADJ\ NOM\ *ADJ)\\DET = [Cat = DET]\\NOM = [Cat = NOM]\\ADJ = [Cat = ADJ]\end{array}\right]\\\left[\begin{array}{l}Cat = GN\\Forme = (PRON)\\PRON = [Cat = VERBE1]\end{array}\right]\end{array}\right.\\GV = \left[\begin{array}{l}Cat = VERBE\\Forme = (VERBE)\\VERBE = [Cat = VERBE]\end{array}\right]\\\left[\begin{array}{l}Catégorie = Phrase\\Forme = (GN\ GN\ GV)\end{array}\right]\end{array}\right]\right\}$$

Figure 1.9 : Description de règles de grammaires (emprunté à [SABAH])

N.B. : Les constituants déjà décrits ne sont pas répétés.

Le formalisme permet également de représenter la structure des phrases à partir des règles de grammaire pertinentes et des représentations des mots de la phrase. Par exemple, pour le GN *le bureau* on a la description reproduite à la figure 1.10 :

$$GN = \begin{bmatrix} Cat = GN \\ Forme = (DET\ \text{NOM}) \\ DET = \begin{bmatrix} Cat = Article \\ Type = Défini \\ Nombre = Sing. \\ Genre = Masculin \\ Lex = le \end{bmatrix} \\ NOM = \begin{bmatrix} Cat = NOM \\ Nombre = Sing. \\ Genre = Masculin \\ Lex = bureau \end{bmatrix} \end{bmatrix}$$

Figure 1.10 : Description d'un groupe nominal (emprunté à [SABAH])

Il est évidemment possible, grâce à l'utilisation des accolades, de conserver l'ambiguïté de certaines phrases.

Le mécanisme par lequel se fait la construction de représentation interne des phrases, s'appelle *unification* mais comme il ne s'agit pas de la même chose que le mécanisme informatique du même nom (bien qu'en étant proche), on préfère le terme *superposition*. La superposition vise à construire une DF qui synthétise deux autres DF, en particularisant éventuellement certains points. On vérifie d'abord que les deux DF à superposer ne présentent pas de contradictions (présence de deux attributs identiques avec des valeurs incompatibles). On construit alors la DF sommaire des attributs présents dans les DF initiales, en donnant les valeurs spécifiques aux attributs communs. La superposition peut en particulier servir à résoudre des ambiguïtés lexicales comme le montre la figure 1.11 :

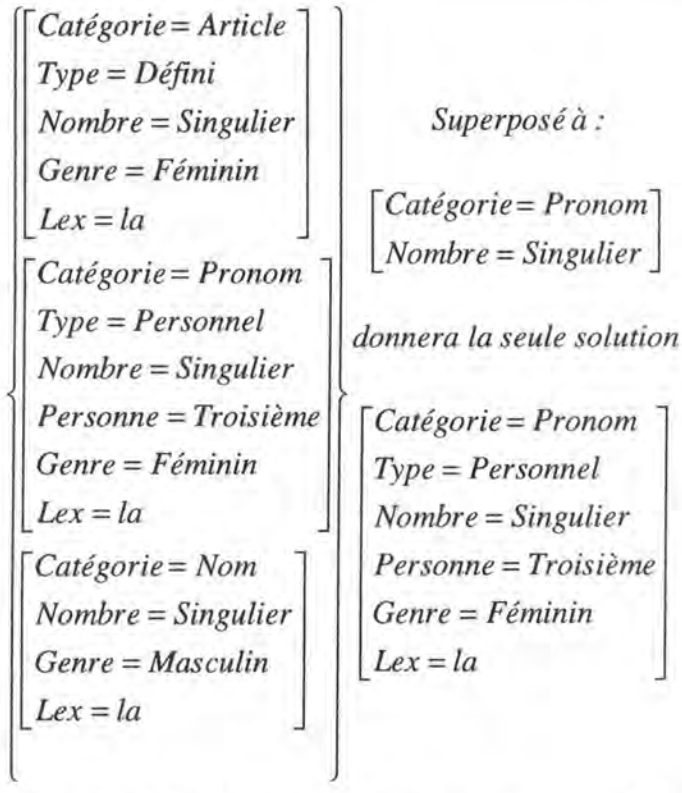


Figure 1.11 : Résolution d'ambiguïtés par superposition (emprunté à [SABAH])

A ce niveau, ce formalisme n'a que la puissance des grammaires formelles de type 2 même s'il est plus pratique et plus général d'utilisation. On a donc trouvé plusieurs moyens d'augmenter sa puissance, notamment par la description de "chemins". Ceux-ci permettent des références multiples à divers éléments des structures décrites par le langage. Ils sont utilisés pour expliciter des contraintes qui, elles, préciseront des notions comme l'accord ("le nom et l'adjectif s'accordent en genre et en nombre") ou les liaisons entre syntaxe et sémantique ("l'agent du prédicat est le sujet syntaxique").

Pour conclure, disons que l'unité qui existe dans les descriptions ainsi que la faculté d'intégration des divers types de connaissances sont à l'origine du grand succès actuel de ce formalisme. Soulignons également que bien ne couvrant pas la même chose que l'unification en programmation (la principale différence étant que l'unification linguistique ignore l'ordre des couples attribut-valeur, l'unification linguistique présente des caractéristiques qui en facilite l'implémentation grâce à des langages de programmation logique.

3.7. Grammaires lexicales fonctionnelles

Dans les grammaires lexicales fonctionnelles (GLF), la description d'une phrase se compose de deux éléments :

- la *structure de constituants* (c-structure), qui traite les aspects syntaxiques et qui est une analyse classique de la phrase par une grammaire indépendante du contexte mais qui autorise un éventail très large de structures syntaxiques même incorrectes (celles-ci seront filtrées dans la f-structure).
- la *structure fonctionnelle* (f-structure), qui traite les aspects sémantiques et est fondée sur la notion de schéma (voir point 3.6 ci-dessus). Il s'agit d'un formalisme très semblable à celui des DF (voir point 3.6 ci-dessus) et où l'attribut FORME n'est pas utilisé puisqu'il relève de la syntaxe.

Aux règles de grammaire de la c-structure, on peut associer des équations garce aux signes ↑ ("up") and ↓ ("down") qui désignent respectivement la f-structure du noeud père et la f-structure du noeud fils. Les règles de la c-structure servent à construire la f-structure.

Exemple d'une telle règle

GN → det adj n
(↑det=↓ (↑mod=↓) (↑=↓)
↑def=↓def
↑num=↓num)

Cette règle affirme que lors de la construction de la f-structure d'un groupe nominal, sa détermination, sa définition et son nombre sont les mêmes que ceux du déterminant qu'il contient. La valeur de modalité est prise à l'adjectif et l'entièreté des couples "attribut-valeur" du nom est copiée dans la f-structure du groupe nominal.

La production d'une phrase par une GLF suit trois étapes :

- à l'aide de la grammaire non contextuelle et en négligeant les équations attachées aux règles, on produit un arbre de dépendances dont les branches sont des catégories lexicales.
- on complète chaque feuille par un mot approprié du dictionnaire (insertion lexicale) et on instancie les équations.
- la f-structure est construite par résolution des équations.

Nous avons vu comment on représente des règles de grammaire dans une GLF. Tout comme dans les grammaires d'unification, il est possible de représenter les entrées lexicales.

Exemple

donne Verbe (↑Temps)=présent, (↑prédicat)=donner C(↑S), (↑O), (↑O2).

Cet exemple est très important car il montre que les GLF autorisent l'intégration dans le lexique d'informations concernant la forme que doit prendre la phrase à un niveau logique (ici un prédicat avec deux sujets et un objet). Il s'agit en fait de données sémantiques.

Enfin, on représente une phrase (ou, ici un groupe nominal) à la manière de l'exemple suivant où sont mêlées c-structure et f-structure.

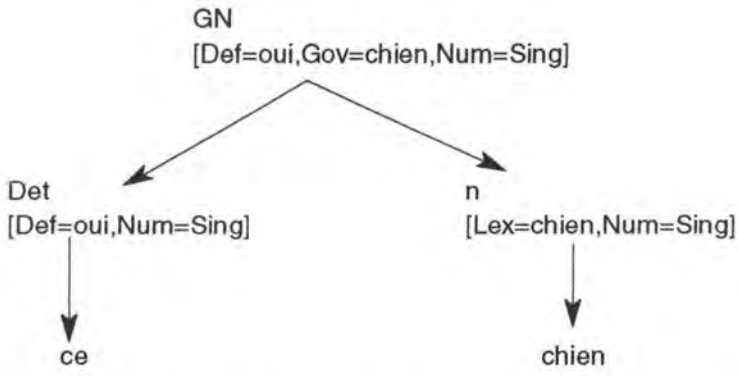


Figure 1.9 : c-structure et f-structure de *ce chien* (emprunté à [SABAH])

Pour conclure sur une évaluation des GLF, rappelons qu'étant basées sur un formalisme semblable aux grammaires d'unification, elles permettent une écriture aisée et une utilisation conjointe de divers types de connaissance (lexicales, syntaxiques, sémantiques, éventuellement pragmatiques). Certains ajouts permettent le traitement relativement aisé des constituants discontinus.

Chapitre II : Traduction Automatique — Cadre général

1. *INTRODUCTION*

Le terme **Traduction Automatique** est l'équivalent français du terme anglais **Machine Translation** que [HUTCHINS] définit comme suit : *"The term Machine Translation (MT) is now the standard name for computerized systems responsible for the production of translation from one natural language into another, with or without human assistance. [...] The term does not include computer-based translation tools which support translators by providing acces to dictionaries and remote technology databases, facilitating the transmission and reception of machine-readable texts, or interacting with word processing, text editing or printing equipment. It does, however, include systems in which translators or other users assist computers in the production of translations, including various combinations of text preparation, on-line interactions and subsequent revisions of output."*

Une vue classique des choses place la traduction humaine — c-à-d effectuée entièrement par l'homme sans recours à un outil informatique quel qu'il soit — et la TA aux deux extrêmes d'un spectre des méthodes de traduction classées selon le mode de coopération entre l'homme et la machine (voir figure 2.1).

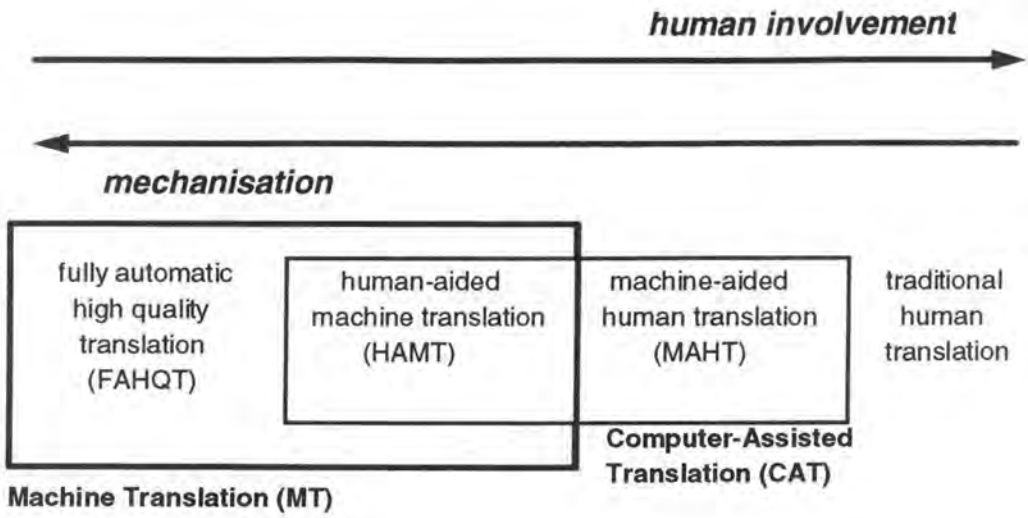


Figure 2.1. : Méthodes de traduction (adapté de [HUTCHINS])

A l'extrême gauche, on trouve la **FAHQT** (*Fully Automatic High Quality Translation*), terme dont Yehoshua Bar-Hillel est à l'origine, qui vise à effectuer des traductions d'une qualité comparable à celle d'un traducteur humain mais sans aucune intervention humaine dans le processus de traduction. Cependant, dans l'état actuel des choses, la FAHQT n'a jamais pu être réalisée et nombreux sont ceux qui pensent, à l'instar de Bar-Hillel, qu'elle restera à jamais un mythe.

Il convient dès lors d'examiner deux autres méthodes de traduction, dites de *Computer-Assisted Translation* (**CAT**) qui sont effectivement mises en oeuvre à l'heure actuelle : la **MAHT** (*Machine-Aided Human Translation*) et la **HAMT** (*Human-Aided Machine Translation*). Il est parfois difficile de dire qu'un système relève de l'une ou de l'autre de ces méthodes mais, globalement, on peut affirmer que la MAHT est l'utilisation d'outils d'informatiques comme aides pour des traducteurs professionnels alors que la HAMT couvre l'utilisation de systèmes de TA afin de produire des traductions avec l'assistance d'un opérateur humain avant, pendant ou après les traitements informatisés.

Pour qu'un outil informatique puisse entrer dans le champs de la MAHT, on considère en général qu'il doit fournir un minimum d'aide linguistique. Un simple traitement de texte n'en fait donc pas partie mais, par contre, on peut y retrouver des outils tels que des correcteurs d'orthographe, des correcteurs grammaticaux, des dictionnaires bilingues "on-line", ...

La HAMT regroupe quant à elle la quasi totalité des systèmes de TA actuels qui, pour qu'on puisse les utiliser en pratique, requièrent qu'un opérateur humain leur facilite la tâche en fournissant une entrée simplifiée, en intervenant dans le processus de traduction lui-même afin de supplanter le système lorsque celui-ci ne peut mener à bien sa tâche ou en agissant sur la

sortie pour rendre celle-ci correcte. Les systèmes restants relèvent en fait de la FAHQT, mais on considère qu'ils ne sont pas visés par les arguments qui prônent l'impossibilité de cette méthode car ils se contentent de traduire des textes fortement limités du point de vue du domaine du discours, du vocabulaire et des types de structures employés.

Si l'on rapproche la classification ci-dessus de la définition donnée au début de l'introduction de ce chapitre, on en déduit que ce qui est visé par le terme TA, ce sont en fait la HAMT et la FAHQT, la FAHQT étant le but ultime de la TA puisque, comme le dit Hutchins: *« [...] the central core of MT is the automation of the full translation process ».*

Ayant ainsi tenté de définir ce que l'on entend par TA, nous allons, dans la suite de ce chapitre, retracer un bref historique de ce domaine (voir section 2) en citant les principaux projets qui ont eu lieu et que nous développerons dans la troisième partie du mémoire. Dans la section 3, nous verrons quelles sont les caractéristiques de base des systèmes de TA. Ces caractéristiques nous permettrons de faire l'analyse des différents systèmes que nous verrons dans la deuxième et la troisième partie du mémoire. Enfin, dans la section 4, nous tentons de situer la TA par rapport à l'Intelligence Artificielle (IA).

2. *BREF HISTORIQUE DE LA TRADUCTION AUTOMATIQUE*

Si l'on veut remonter aux toutes premières idées qui sont susceptibles d'avoir préparé le terrain pour ce qui, plus tard, est devenu la TA, il faut sans doute remonter à Descartes et Leibniz qui, au 17ème siècle, avaient envisagé la création de dictionnaires se basant sur des codes numériques universels. L'idée générale dans laquelle venait s'inscrire cette proposition était celle de l'existence d'une langue universelle à travers laquelle toute l'humanité pourrait communiquer.

Un peu plus près de nous, en 1933, la première tentative de mécaniser un travail de traduction est attribuée parallèlement au Franco-Arménien George Artsouni et au Russe Petr Smirnov-Troyanskii. Les travaux de ces deux chercheurs restèrent méconnus car les circonstances de l'époque ne s'y prêtaient pas.

Généralement, on considère que se sont Warren Weaver et Andrew Donald Booth qui ont commencé à susciter l'intérêt du monde scientifique pour l'idée qu'un ordinateur pourrait servir à effectuer des traductions d'une langue vers une autre. Ils se doutaient que la seule

automatisation des recherches dans un dictionnaire ne pouvait pas, à elle seule, régler tous les problèmes. Les problèmes qui restaient à régler étaient d'après eux :

- l'existence de plusieurs traductions pour un même mot en fonction du contexte,
- la variation de l'ordre des mots d'une langue à l'autre,
- la nécessité de traduire comme un tout les expressions idiomatiques.

Cependant, pour eux, le principal problème de la TA reste un problème de vocabulaire et ils furent à la base de l'approche directe (voir section 3.2 de ce chapitre), que l'on peut qualifier grossièrement de traduction "mot-à-mot" et qui fut celle de la première génération des systèmes de TA.

En 1947, Booth et D.H.V. Britten ont réalisé le premier programme de consultation de dictionnaire. Suite aux suggestions de R.H. Richens, Booth se mit à étudier avec celui-ci la possibilité d'automatiser la consultation d'un dictionnaire dans lequel on n'aurait pas une entrée pour chaque mot mais bien pour chaque forme canonique d'un mot. Cela réduit fortement la taille du dictionnaire mais nécessite l'ajout au programme de règles d'inflexions des mots (par exemple, pour produire "loves", "loving", ... à partir de "love" lors de l'analyse du texte source et retrouver "love" à partir de "loves ", "loving", ... lors de la génération du texte cible).

Dans un memorandum paru en juillet 1949, Weaver propose diverses méthodes inspirées de découvertes récentes de l'époque : l'utilisation de techniques cryptographiques développées pendant la guerre (domaine dans lequel travaillaient Weaver et Booth), l'analyse statistique, la théorie de l'information de Shannon, l'exploration de la logique sous-jacente au langage ainsi que de ses caractéristiques universelles.

Quatre idées importantes ressortent de ce memorandum :

- tout d'abord, l'idée générale que, toutes les langues ont un grand nombre de caractéristiques communes;
- il existe, pour une occurrence d'un mot à l'intérieur d'une phrase, une "fenêtre" de 2N+1 (où N est fonction du mot) mots du texte qui suffit pour déterminer de manière univoque (avec un risque d'erreur très faible) la traduction adéquate du mot;
- le texte source et le texte traduit signifient la même chose : il suffit d'un (dé)codage pour passer de l'un à l'autre;

- traduire d'une langue à une autre signifie d'abord passer par une "langue universelle" intermédiaire (appelée communément *interlingua*) hypothétiquement "connue" de tous.

Cette dernière idée donna lieu à celle d'une représentation unique de la sémantique et on la retrouve dans le domaine de l'IA qui s'intéresse à la représentation des connaissances. C'est elle également qui donna naissance à l'approche indirecte et, plus particulièrement, à sa première variante : l'*interlingua* (voir point 3.2 de ce chapitre).

Suite à la parution du memorandum, la recherche en TA s'intensifia fortement. En 1951, Yehoshua Bar-Hillel, le premier chercheur rémunéré en TA, organisa la première conférence de TA lors de laquelle nombre de propositions marquantes furent déjà émises au niveau du traitement de la syntaxe, des restrictions sur les textes à traduire, de la nécessité d'une intervention humaine (pré- et post-édition) en attendant une automatisation complète.

Durant les années suivantes, de nombreux projets furent lancés, les uns se basant sur des théories (existantes ou développées pour le projet), les autres travaillant par "essai-erreur". Cette période fut prolifique en apports scientifiques tant en TA qu'en linguistique et en IA. Cependant, le fait est que personne ne réussit à mettre au point un système produisant des traductions de "bonne qualité". La plupart des systèmes de l'époque utilisaient une approche directe ou une approche *interlingua* basée sur la syntaxe (voir point 3.3 de ce chapitre), mélangeaient données linguistiques et programmes dans leur code et manquaient généralement de cohérence du point de vue de leur modèle linguistique. C'était donc la réalisation pratique des systèmes de TA qui fut mise momentanément en doute.

Mais, d'un point de vue théorique cette fois, Bar-Hillel en vint à penser que la FAHQT ne pourrait jamais être réalisée. Pour lui, la quantité et la diversité des connaissances dont doivent disposer les systèmes de TA pour résoudre les ambiguïtés syntaxiques et sémantiques sont trop importantes. Il affirme : *"A translation machine should not only be supplied with a dictionnary but also with a universal encyclopedia"*. (Nous développerons quelque peu l'argument de Barr-Hillel à la section 4 de ce chapitre, où nous tentons de situer la TA par rapport à l'IA).

L'avis de Bar-Hillel ne fut pas partagé par tous mais il eut une très forte influence et il influença certainement le point de vue officiel du gouvernement américain qui participait activement au financement de la recherche en TA. Ce dernier avait créé l'ALPAC (*Automatic Language Processing Adivisory Committee*) qui, dans son célèbre rapport de 1966, a conclu que la TA était plus lente, moins précise et beaucoup plus coûteuse que la traduction humaine et qu'il n'y

avait pas d'avenir immédiat dans ce domaine. Cela mit fin aux investissements gouvernementaux et engendra une baisse des prétentions dans le domaine de la TA.

L'approche *interlingua* paraissant utopique, la plupart des systèmes qui virent le jour par la suite, adoptèrent une méthode (moins difficile à mettre en oeuvre) dite de *transfert* (voir 3.2 ci-dessous). Cette méthode a l'avantage de ne pas nécessiter le passage par une représentation intermédiaire "universelle" du texte. Ces projets se créèrent surtout en dehors des Etats-Unis : au Canada et en Europe de l'Ouest, lieux où cohabitent plusieurs communautés linguistiques et où la TA présente dès lors un attrait plus important. Le rapport ALPAC marqua donc une véritable rupture et c'est dans la période qui suivit cette rupture que furent lancés les projets les plus marquants et dont beaucoup sont encore d'actualité. Ceux que nous examinerons dans la suite du travail sont repris ci-dessous.

1976 vit la création par un groupe de recherche installé à Montréal du système **TAUM-Meteo** (voir chapitre IX) que l'on considère comme l'archétype des systèmes utilisant un sous-language (voir point 3.6 du chapitre II). Ce système a pour but de traduire des bulletins météorologiques de l'anglais vers le français afin d'être diffusés.

La même année, la Commission des Communautés Européennes décida de faire développer **Systran** (voir chapitre VII), un système anglais-français, sur base d'un autre système qui était utilisé par l'US Air Force depuis 1970 pour des traductions russe-anglais. A l'anglais-français vinrent ensuite s'ajouter d'autres paires de langues telles que français-anglais, anglais-italien, anglais-allemand, etc.

A la fin des années 70, la Commission décida également de financer le projet **Eurotra** (voir chapitre V). Ce projet, disposant de groupes de recherche dans tous les états membres, était très ambitieux et faisait usage des percées les plus récentes en TA et en linguistique en vue de la réalisation d'un système multilingue. Pour sa conception, Eurotra s'inspira de deux autres systèmes : Ariane, un système russe-français de type *interlingua* développé à Grenoble pendant les années 60 et **Susy** (voir chapitre VI), un système multilingue de type transfert développé à Saarbrücken depuis la fin des années 60.

A cette époque, il apparu que des projets significatifs ne pouvaient se faire que grâce à une approche de type *transfert*. C'est d'ailleurs en partageant ce point de vue que fut lancé le projet **Metal** (voir chapitre X) à Austin (Texas).

Depuis les années 80, on constate des utilisations conjointes des approches *interlingua* et *transfert* suite à la constatation que les connaissances linguistiques seules ne suffisent pas à la

traduction, qu'une "compréhension" du texte est nécessaire et que, pour cela, des connaissances (extra-linguistiques) sur le monde sont nécessaires.

Ces dernières années ont vu l'arrivée de projets traitant de la Traduction Automatique de la parole, chose qui, jusque là, était impensable.

Mais le fait le plus marquant de ces dix dernières années est l'apparition des systèmes de TA commerciaux dont la qualité du point de vue linguistique est loin d'être irréprochable mais qui arrivent à fournir des produits qui, dans des circonstances bien précises, se révèlent d'une grande utilité. Pour cela, on réalise des adaptations du produit en fonction du client. On fait aussi très souvent appel à la pré- et à la post-édition des textes soumis au système.

Dans les systèmes commerciaux on retrouve principalement Metal (dont le développement a été repris par Siemens), Systran ainsi que le système américain **Logos** qui fait l'objet d'une étude approfondie dans la deuxième partie du travail.

L'état actuel de la TA se caractérise donc par une multiplicité des types de systèmes et par la constatation que le noeud du problème n'est pas du côté de la technologie mais de celui du langage et de sa formalisation, préalable nécessaire à son traitement par la machine. C'est pourquoi, à l'heure actuelle, de nombreux champs de recherche sont encore ouverts tant en linguistique théorique qu'en IA.

3. *CARACTERISTIQUES DE BASE DES SYSTEMES DE TA*

Dans cette section, nous détaillons les aspects des systèmes de TA qui nous semble les caractériser le mieux. Ceux-ci nous servirons de point de départ à la critique de Logos (chapitre 4) et des autres systèmes que nous verrons dans la troisièmes partie. Ci-dessous, nous allons examiner le choix d'un modèle linguistique, celui du nombre de langues que le système peut accepter, la stratégie du système (directe ou indirecte, et, parmi les système indirects, *transfert* ou *interlingua*), le degré d'interactivité, l'organisation des données lexicales et la restriction à un sous-langage, l'utilisation pratique du système et nous terminerons en faisant état de quelques caractéristiques liées à l'implémentation.

3.1. L'approche linguistique

Les modèles linguistiques les plus fréquemment utilisés en TA ont été passés en revue au chapitre précédent. Ici, nous insistons seulement sur le fait que les systèmes de TA actuels ne se basent pas en général sur une seule théorie linguistique. La plupart du temps, les systèmes adoptent une approche générale (telle qu'une grammaire transformationnelle ou un représentation par arbres de dépendances) mais la modifient fortement, d'une part, par des emprunts à d'autres théories et, d'autre part, pour des besoins d'implémentation.

Par ailleurs, une des critiques les plus saillantes subies par la TA est celle d'ignorer les développements des théories linguistiques. Il n'est donc pas sûr que les systèmes de TA aient toujours les fondements théoriques les plus récents et les plus cohérents.

Les questions qu'il est intéressant de se poser au sujet de l'approche linguistique adoptée par un système sont principalement :

- Quels sont les niveaux linguistiques traités et comment le sont-ils ? Par *comment*, on entend surtout le fait de savoir si, dans la phase d'analyse, les nivaux sont traités séparément ou si des informations de différents niveaux sont accessibles au même moment afin de résoudre certaines ambiguïtés. On pense surtout à la disponibilité en même temps d'informations syntaxiques et sémantiques, ce qui s'est depuis longtemps révélé être une nécessité.
- Comment caractériser la puissance d'expression (théorique et pratique) du formalisme employé ?
- Quels sont les niveaux de généralité des différents types de règles linguistiques que l'on trouve dans le système ? Il faut se poser la question dans le but de repérer les situations où il est fait usage de traitements *ad hoc* et où il aurait été plus opportun d'avoir un traitement plus général.

3.2. Un système bilingue ou multilingue

Un système **bilingue** est un système permettant d'effectuer des traductions entre deux langues. Un système **multilingue** permet d'effectuer des traductions entre plus de deux langues.

Les systèmes bilingues peuvent être qualifiés d'**uni-directionnels** ou de **bi-directionnels** selon qu'ils peuvent effectuer des traductions dans un sens uniquement ou dans les deux. Les systèmes bi-directionnels sont qualifiés de **réversibles** si le processus d'analyse du texte dans

une langue donnée peut être utilisé "à l'envers" pour produire une traduction dans cette même langue. Les systèmes de ce dernier type étant très difficiles à mettre au point, tant en théorie qu'en pratique, la plupart des systèmes bilingues sont, en fait, constitués de deux systèmes uni-directionnels de sens opposés.

Dans un système multilingue, il y a une série de langues sources et une série de langues cibles, mais les traductions dans toutes les paires de langues n'existent pas forcément. Dans un système **"réellement" multilingue**, les composantes d'analyse et de génération relatives à une langue donnée restent les mêmes quelle que soit l'autre langue concernée (ce qui implique également l'utilisation d'une même approche linguistique). En caricaturant un peu, on peut dire que, parmi les système multilingues, il y a, d'une part, les systèmes réellement multilingues et, d'autre part, ceux qui sont plus une collection de systèmes bilingues.

L'avantage des systèmes réellement multilingues est qu'ils permettent la réutilisation des composantes d'analyse et de génération des langues engendrant ainsi une plus grande modularité du programme. Leur désavantage est qu'ils ne permettent pas d'exploiter les ressemblances accidentelles (de vocabulaire et de structure) qui existent entre deux langues.

3.3. La stratégie du système

Les premiers systèmes de TA adoptaient ce que l'on appelle une **approche directe** qui peut être représentée par le schéma de la figure 2.2.

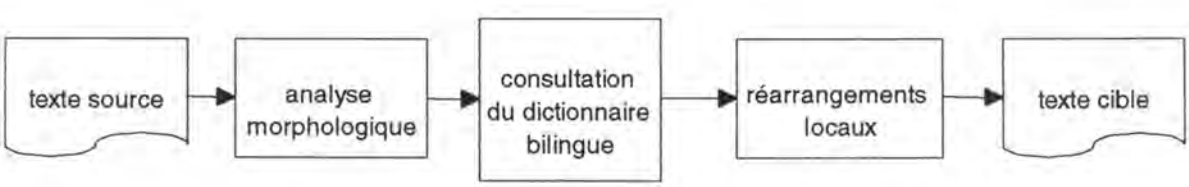


Figure 2.2. : Systèmes directs (emprunté à [HUTCHINS])

Dans ces systèmes bilingues, à une analyse morphologique ayant pour but de trouver les formes non fléchies des mots, succède une consultation du dictionnaire, laquelle, moyennant quelques réarrangements locaux (par exemple, la permutation des structures nom/adj en adj/nom pour les traductions anglais-français), mène directement à la traduction. Ce type de systèmes se caractérise par le manque d'étapes intermédiaires (analyse et génération) dans le processus de traduction et ne permet en fait que d'effectuer des traductions "mot-à-mot". Il n'y a pas d'analyse syntaxique du texte source, ni de représentation interne de sa sémantique. Cette approche, qui fut celle de la première génération des systèmes de TA, est apparue assez tôt

comme naïve et la deuxième génération de système de TA adopta l'approche indirecte développée ci-dessous. Notons cependant que l'approche directe est encore parfois utilisée dans une certaine mesure dans des systèmes bilingues voulant exploiter les similarités de vocabulaire et de structure entre deux langues.

L'**approche indirecte** a deux variantes principales : transfert et interlingua.

L'idée de l'approche **interlingua** est que l'analyse du texte source donne lieu à une représentation intermédiaire contenant toute l'information nécessaire à la génération du texte cible (la génération doit donc se servir uniquement de cette représentation et ne doit plus avoir besoin de consulter le texte source). Dans cette approche (illustrée par la figure 2.3), l'analyse et la génération ne sont relatives qu'à une seule langue et peuvent donc être utilisées quelle que soit l'autre langue concernée. C'est pourquoi l'approche *interlingua* semble la mieux adaptée aux systèmes multilingues. En effet, comme le montre l'exemple de la figure 2.4, dans un système de type *interlingua*, on peut passer de deux paires de langues à six en ajoutant un module d'analyse et un module de génération pour une langue supplémentaire (l'allemand en l'occurrence).

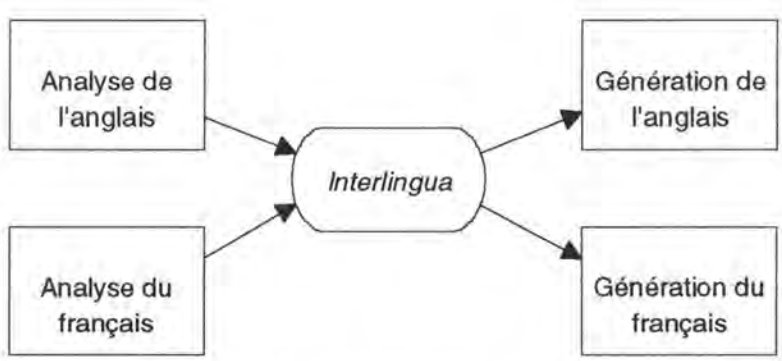


Figure 2.3. : Système *interlingua* avec l'anglais et le français, tous deux comme langues sources et cibles (emprunté à [HUTCHINS])

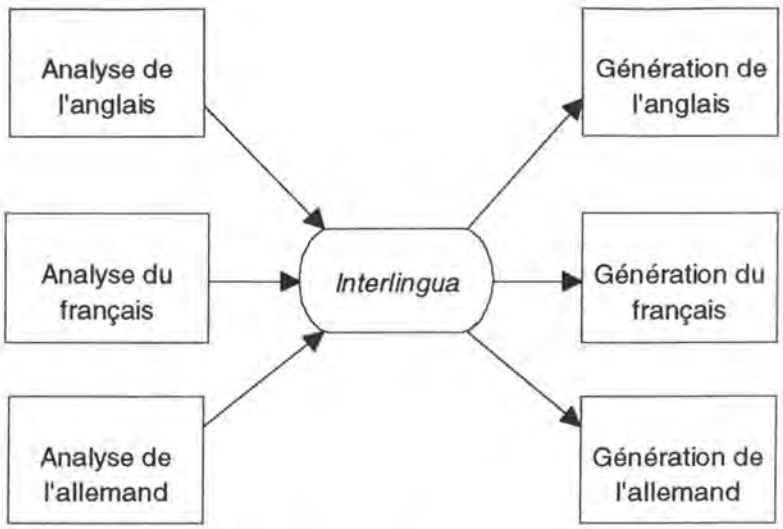


Figure 2.4. : Système *interlingua* avec l'anglais, le français et l'allemand, tous trois comme langues sources et cibles (emprunté à [HUTCHINS])

L'inconvénient de l'approche *interlingua* est d'ordre théorique. Il s'agit en fait de la difficulté de définir une représentation intermédiaire qui soit vraiment "universelle".

Une des premières propositions fut d'utiliser une langue naturelle comme *interlingua*. Mais une telle manière de faire implique une perte supplémentaire liée à la traduction dans une langue intermédiaire. En fait, un tel système revient à avoir deux systèmes : l'un entre les langues sources et la langue intermédiaire, l'autre entre la langue intermédiaire et les langues cibles.

D'après [TUCKER], les approches qui peuvent réellement être qualifiées d'*interlingua* se divisent en deux classes : l'approche syntaxique et celle qui est inspirée de l'IA. L'approche syntaxique qui, historiquement, précéda l'approche *transfert*, est basée sur l'idée, issue de Chomsky, que la structure syntaxique du texte en langue source peut être considérée comme universelle et donc être utilisée directement par la phase de génération. Mais la syntaxe (même au niveau de la structure profonde) ne semble pas être suffisante pour faire office d'*interlingua*. On en est donc arrivé à penser que la meilleure façon de mettre en oeuvre cette approche est d'avoir une représentation intermédiaire capable de refléter la sémantique du texte. Les systèmes de ce type sont ceux qui s'inscrivent également dans le cadre de la recherche en IA mais ils ne constituent qu'une petite partie des systèmes de TA. Leur fonctionnement est illustré à la figure 2.5 et est résumé comme suit par [CARBONELL] : *« First, the source text is analysed and mapped into a language-free conceptual representation. Inference mechanisms then apply contextual world knowledge to augment the representation in various ways, adding information about items that were only implicit in the*

text. Finally, a natural-language generator maps appropriate sections of the language-free representation into the target language. ».

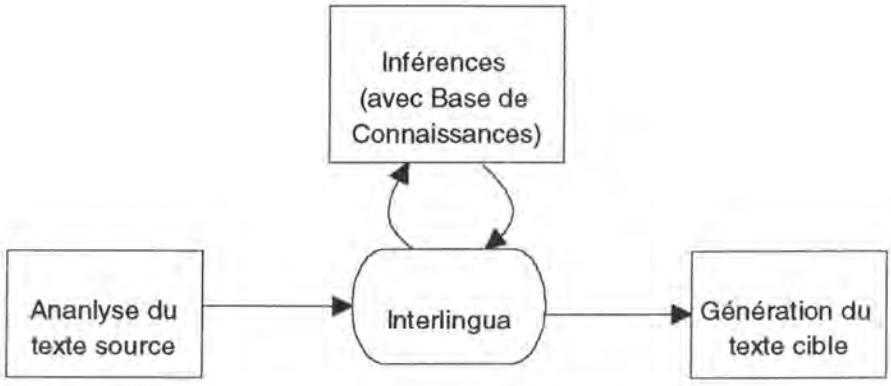


Figure 2.5. : Système *interlingua* avec inférence sur la représentation interne (approche IA)

La deuxième variante de l'approche indirecte est connue sous le nom de méthode de **transfert**. Il s'agit de l'approche utilisée par les systèmes qui passent par un module bilingue (module de transfert) entre la représentation intermédiaire du texte source (fournie par l'analyse) et la représentation intermédiaire du texte cible (utilisée par la génération). Ces deux représentations sont des abstractions des textes sources et cibles mais restent dépendantes de la langue dans laquelle sont écrits ces textes. Le module de transfert traduit la représentation du texte source en une représentation équivalente du texte cible. Le schéma général des systèmes de type *transfert* est représenté à la figure 2.6 ci-dessous :

Figure 2.6. : Systèmes de type *transfert*

On estime généralement ([TUCKER]) que, pour l'application de l'approche *transfert*, deux choix interdépendants sont à effectuer :

- l'importance relative des parties monolingues (analyse et génération) par rapport à la partie bilingue (transfert) du processus. Il s'agit en fait du niveau et de la précision des connaissances linguistiques de la représentation produite par l'analyse et de la représentation de laquelle part le module de génération.
- le niveau de transfert. Lorsque les représentations intermédiaires des textes source et cible sont assez élevées pour exprimer des caractéristiques des textes communes aux deux langues concernées (comme c'est, par exemple, le cas pour le système Eurotra - voir chapitre V - qui utilise des représentations internes au niveau de la

structure profonde), le module de transfert peut se limiter à simplement rechercher, dans un lexique bilingue, les équivalents des mots de la langue source. On parle dans ce cas de transfert à un niveau lexical, c'est-à-dire à un niveau relativement peu élevé. Si les représentations intermédiaires sont de niveau linguistique moins élevé et ne font donc intervenir que peu de caractéristiques "universelles" des langues, le module de transfert devra effectuer des traitements plus importants pour convertir la représentation du texte source en celle du texte cible et devra donc faire utilisation de connaissances linguistiques plus importantes. Le niveau de transfert sera alors relativement plus élevé.

Dans un système de type *transfert*, l'ajout d'une langue comme source et cible requiert un module d'analyse et un module de génération pour cette langue. Mais il faut en plus qu'il y ait un module de transfert pour chaque paire de langues dans laquelle on veut pouvoir effectuer des traductions. Dans un environnement multilingue, l'approche *transfert* souffre donc d'une relative inefficacité. Mais, d'un point de vue théorique, l'approche *transfert* ne pose pas le problème de la nécessité d'une représentation intermédiaire "universelle" et c'est pourquoi elle est souvent préférée à l'approche *interlingua*.

3.4. Les systèmes non-interventionnistes et les systèmes interactifs

La caractéristique dont il est ici question se réfère au rôle joué par l'utilisateur du système.

Les systèmes **non-interventionnistes** sont ceux qui ne requièrent aucune intervention d'un opérateur humain lors du processus de traduction une fois le texte source fourni. L'origine de cette façon de faire se situe dans le traitement "batch" qui était la norme aux débuts de l'informatique. Beaucoup des systèmes actuels ont été conçus lorsque l'informatique ne connaissait pas encore d'interaction homme-machine sophistiquée et sont donc non-interventionnistes.

Les systèmes **interactifs**, par contre, demandent l'intervention de l'utilisateur lors de la traduction afin de mener à bien celle-ci.

Ces systèmes peuvent avoir besoin d'un supplément d'information linguistique, de la confirmation d'une décision ou bien d'un choix parmi plusieurs possibilités. Il faut cependant savoir ce qu'on peut demander à l'utilisateur selon que celui-ci connaît la langue cible, la langue source ou bien les deux.

3.5. L'organisation de l'information lexicale

Grossièrement, on peut distinguer dans les systèmes de TA, information lexicale et information grammaticale. Cette dernière est celle qui est contenue dans les grammaires servant à l'analyse et à la génération. Par information lexicale, par contre, on entend l'ensemble de l'information attaché à chaque élément lexical (mot ou groupe de mots) qui fait partie du vocabulaire d'une langue donnée. Dans la plupart des systèmes, information lexicale et grammaticale sont séparées.

L'information lexicale est rassemblée dans des lexiques qui sont parfois aussi appelés dictionnaires mais dont le contenu diffère assez fortement de ce qu'on retrouve dans ce qu'on appelle communément un dictionnaire. Une entrée de lexique (qui peut être une racine ou la forme fléchie complète) a pour but de donner de l'information pour les traitements syntaxiques et sémantiques. C'est pourquoi on y retrouve des renseignements tels que la catégorie syntaxique, des informations morphologiques, les valences, les "case frames ", des informations sémantiques, des restrictions quant à leur sélection, etc.

La stratégie (voir point 3.3 de ce chapitre) et l'interactivité (voir point 3.4 de ce chapitre) du système ont une incidence sur l'organisation des données lexicales. Dans un système direct, on retrouvera généralement un lexique bilingue, c'est-à-dire contenant des informations sur les items lexicaux de la langue source ainsi que sur leurs équivalents dans la langue cible. Pour chacun des équivalents, il faudra retrouver de l'information nécessaire pour faire le choix parmi les différentes alternatives ainsi que des données syntaxiques qui permettront de réordonner les mots dans le texte cible. Il en résulte que dans les systèmes directs, les lexiques peuvent être très complexes.

Dans les systèmes indirects par contre, les phases d'analyse et de génération étant indépendantes, on retrouve des lexiques monolingues distincts pour les langues source et cible ainsi qu'un lexique bilingue servant au transfert.

Le lexique utilisé par l'analyse doit contenir l'information nécessaire à l'analyse syntaxique du texte et à la levée des ambiguïtés d'homographes et de polysèmes. Souvent d'ailleurs, les homographes et les polysèmes ont une entrée distincte pour chacune de leurs significations. Un lexique de transfert peut être très simple puisqu'il ne doit servir qu'à établir une correspondance lexicale entre item de la langue source et item de la langue cible (par exemple, au terme anglais *ban*, correspondent les termes français *banque* et *rive*). Enfin, le lexique utilisé par la phase de génération est souvent plus simple que celui utilisé par l'analyse puisqu'il ne doit pas permettre de lever les ambiguïtés d'homographes et de polysèmes.

Dans la pratique, les choses sont un peu plus compliquées car les lexiques sont très souvent scindés en "sous-lexiques" propres à un certain domaine (biochimie, métallurgie,...) dans le but de diminuer les problèmes d'homographie.

Il y a aussi le fait qu'au lexique principal, on choisit parfois d'ajouter quelques lexiques spécialisés (par exemple pour les termes très fréquents, les expressions idiomatiques, les formes irrégulières, ...) dont la consultation est réalisée par des parties spécialisées du système.

Nous venons de voir ci-dessus l'influence du type (direct, *transfert* ou *interlingua*) d'un système sur ses données lexicales mais le mode d'interaction avec l'utilisateur peut également jouer un rôle. Par exemple, dans un système interactif où le système requiert l'intervention de l'utilisateur pour choisir entre plusieurs équivalents, dans la langue cible, d'un même mot de la langue source, ou pour résoudre des ambiguïtés d'ordre grammatical ou des ambiguïtés d'homographie, les informations attachées à un item lexical ne doivent plus contenir d'information complexe tant syntaxique que sémantique. A la limite, seule une liste d'alternatives devant être soumise à l'utilisateur s'y retrouve. Tout dépend en fait de la part d'"intelligence" que l'on donne au système par rapport à celle qu'on laisse à l'utilisateur.

3.6. La restriction à un sous-langage

Si un système n'est conçu que pour traiter des textes d'un domaine du discours particulier, on dit que le système travaille sur un **sous-langage**. Dans le cas contraire, on dit qu'il s'agit d'un système à usage général (le terme anglais *general purpose* est également couramment employé). Cette approche est née avec le système Météo qui, rappelons-le, effectue des traductions de rapports météorologiques. Elle est utilisée également par d'autres systèmes dans des domaines tels que l'aviation, l'industrie textile, etc.

Ce qui motive cette approche est que de nombreux domaines :

- ont des terminologies bien standardisées;
- bien que des constructions grammaticales leur soient rarement spécifiques, ils en utilisent certaines de manière très fréquente;
- contiennent, dans leur sous-langage, des mots dont les propriétés grammaticales diffèrent de ce qu'elles sont dans la langue usuelle.

Les systèmes de TA conçus pour un sous-langage bénéficient des avantages suivants :

- la réduction des ambiguïtés d'homographie en raison de la restriction à un domaine du discours et à la plus grande standardisation de la terminologie;
- la possibilité de se concentrer sur les problèmes grammaticaux que l'on rencontre fréquemment dans les textes du domaine;
- la destination du système à une tâche spécifique avec la possibilité de mesurer assez clairement l'acceptabilité des sorties du système.

Certains problèmes demeurent cependant. Premièrement, certains sous-langages, en raison de leur grande complexité, présentent plus de difficultés à être traités que la langue usuelle. On pense surtout ici au langage légal. Deuxièmement, dans les faits, on constate qu'il est rare que les textes fassent référence à un seul domaine du discours. De plus, les systèmes conçus pour un sous-langage donné ne sont pas forcément adaptables à d'autres.

3.7. L'utilisation pratique des systèmes de TA

Etant entendu, comme nous l'avons dit dans l'introduction de ce chapitre, que la TA, à l'heure actuelle, relève pour une partie négligeable de la FAHQT et, pour sa quasi-totalité, de la HAMT, on en déduit que, dans l'analyse des systèmes de TA, l'intervention humaine est un critère à prendre en considération. Nous avons déjà parlé des modes de travail interactifs au point 3.4 de ce chapitre. Rappelons que ce que nous appelons "interactif" se réfère à l'intervention humain pendant le processus de traduction. Dans cette section, par contre, nous allons aborder l'intervention humaine avant et après le processus de traduction, c'est-à-dire la pré- et la post-édition.

3.7.1. La pré-édition

[HUTCHINS] la caractérise comme suit : « *Typically, pre-editing involves checking source texts for forseable problems for the system and trying to eradicate them. It can include the identification of names (proper nouns), the marking of grammatical categories of homographs, indication of embedded clauses, bracketing of coordinate structures, flagging or substitution of unknown words, etc. In its extreme form, it involves the reformulation of the text using a 'controlled language'* ».

Ajoutons que la pré-édition peut être **obligatoire** ou **facultative**. Lorsqu'elle est facultative, le système cherche des marquages qui pourraient lui faciliter la traduction mais traduit (sans doute avec une qualité moindre) même s'il ne les trouve pas; il ne peut le faire si la pré-édition est obligatoire.

La méthode la plus "extrême" de pré-édition est l'utilisation d'un **langage contrôlé** dans le texte source. Cette façon de faire trouve son origine dans la constatation que les problèmes majeurs posés à la TA sont l'impossibilité d'interpréter correctement certaines constructions et la présence d'ambiguïtés (surtout liées aux homographes). L'utilisation d'un langage contrôlé a pour but d'adapter le texte source aux constructions et au vocabulaire que le système arrive à traiter de manière satisfaisante. Il est important ici de faire la distinction entre la notion de langage contrôlé et celle de sous-langage que nous avons vu au point 3.6 de ce chapitre. Les différences entre les deux sont que :

- le langage contrôlé, par opposition au sous-langage, n'est pas limité à un domaine du discours;
- les systèmes qui utilisent un langage contrôlé n'ont pas été conçus pour ne fonctionner qu'avec celui-ci, c'est-à-dire qu'ils peuvent traduire des textes non contrôlés même si c'est avec moins de réussite.

Les deux approches peuvent cependant bien évidemment être combinées de manière avantageuse et l'on parle alors de sous-langage contrôlé.

3.7.2. La post-édition

Nous l'avons déjà dit, les traductions produites par les systèmes de TA actuels ne sont pas parfaites. Dans le cas où celles-ci sont destinées à être diffusées, des corrections doivent alors être faites. C'est le but de la post-édition. L'ampleur de celle-ci dépend, d'une part, de la qualité des traductions fournies par le système et, d'autre part, de la qualité que l'on veut atteindre.

Dans les premiers systèmes, aucune aide n'était disponible pour la post-édition. Celle-ci devait être faite à la main et toutes les erreurs devaient être découvertes par l'utilisateur. Ensuite, sont apparues des choses plus évoluées comme la disponibilité en parallèle à l'écran du texte source et du texte cible, la possibilité de réordonner les mots et les propositions sans devoir les réécrire intégralement. A l'heure actuelle, on trouve des systèmes qui signalent à l'utilisateur les parties de phrases qui pourraient être mal traduites, qui offrent la possibilité, après la

correction d'une erreur, de corriger automatiquement toutes les erreurs similaires. Le futur nous laisse même entrevoir des environnements de post-édition linguistiquement "intelligents" qui pourraient proposer des alternatives à la traduction choisie par le système en cas d'ambiguïté, qui feraient automatiquement les changements d'accord, etc.

3.8. Caractéristiques d'implémentation

La caractéristique principale qui permet de différencier les systèmes de TA du point de vue de leur implémentation est la manière dont y sont stockées les données (principalement lexicales et grammaticales). Ces données contiennent véritablement toute l'intelligence du système et il est important de pouvoir y accéder efficacement. Dans les systèmes les plus anciens, il arrive que ces données se retrouvent dans les même parties de code que les algorithmes qui les emploient nuisant ainsi à la clarté et donc aux possibilités de maintenance du système. Cet état de fait est lié à l'absence d'un formalisme déclaratif permettant d'exprimer les données linguistiques. Les systèmes plus récents ont stockés ces données sur un support en mémoire secondaire (donc détachées de la partie algorithmique du système), la majorité utilisant encore un système de fichiers, une petite partie faisant utilisation de la technologie des bases de données. Ces systèmes permettent des modifications plus aisées ainsi qu'une quantité plus importante de données stockées.

Soulignons également l'existence de systèmes "expérimentaux" écrits en LISP ou en PROLOG et qui souvent ne disposent pas d'une gestion efficace des données en mémoire secondaire.

4. LA TA ET L'INTELLIGENCE ARTIFICIELLE

Dans la discussion sur l'automatisation de la traduction des langues, un point de vue fait l'unanimité chez d'éminents auteurs ([WILKS], [SABAH]) : la nécessité d'une compréhension du texte afin de pouvoir le traduire.

Tout d'abord se pose la question de savoir ce que signifie "comprendre le langage". La réponse n'est pas claire pour l'homme, elle l'est d'autant moins pour la machine ! La première chose qu'il nous faut constater, et qu'ont compris les théoriciens de la linguistique, est que l'étude des phénomènes linguistiques parmi les plus fréquents du langage naturel ne peut être faite sans prise en considération de la sémantique de la phrase. Et même, les recherches qui se sont

limitées à la seule phrase ne peuvent rendre compte de tous ces phénomènes. Il est donc clair que le contexte dans lequel la phrase apparaît doit être pris en compte et que le sens d'un texte n'est pas la juxtaposition du sens des phrases qui le composent.

Des tas de questions encore plus ou moins sans réponse sont liées à la question de la compréhension du langage. *Comment le programme peut-il reconnaître que quelque chose n'a pas de sens ? Comment reconnaître une métaphore d'une telle phrase ? Comment interpréter les éléments nouveaux ?* Comprendre le langage signifie aussi le situer par rapport à ses buts. Il est nécessaire de comprendre les buts de l'interlocuteur, d'inférer son plan pour comprendre correctement ce qu'il dit. Un modèle du comportement de l'interlocuteur est nécessaire.

On s'aperçoit vite que les questions et les problèmes soulevés ci-dessus ne trouvent pas de solution dans les modèles linguistiques que nous avons vu au chapitre précédent. Ceux-ci, en effet, traitent du langage au niveau de la phrase et ne prennent en compte que des aspects limités de la sémantique, la syntaxe étant de loin le centre d'intérêt principal de ces modèles. C'est dans le champs d'application de ces modèles que se situent actuellement la grande majorité des projets de TA. Ils font utilisation de modèles linguistiques essentiellement syntaxiques, existants ou *ad hoc*, afin d'effectuer des traductions d'un bon niveau sans que leur système ait une "compréhension approfondie" du texte et donc sans vouloir non plus, en général, simuler la manière dont l'homme effectue des traductions.

On pourrait alors se poser la question du *pourquoi* de cet état de fait. La réponse est sans doute à trouver dans l'argument que formula Barr-Hillel au sujet de l'impossibilité de la FAHQT et qui, comme on l'a vu dans l'historique de la TA (voir section 2 de ce chapitre), eut une grande influence sur la direction des recherches dans ce domaine. Barr-Hillel utilisa, par exemple l'histoire d'enfant suivante : *Little John was looking for his toy box. Finally he found it. The box was in the pen.* Un interlocuteur comprend ici que *pen* ne se réfère pas à un stylo mais qu'il s'agit en fait d'une ellipse pour *playpen* qui signifie "parc à bébé". Il y arrive garce à sa connaissance du monde et aux déductions qu'il peut en faire. Barr-Hillel en conclut que l'automatisation de la traduction *[...] nécessiterait de communiquer non seulement un dictionnaire, mais aussi des connaissances encyclopédiques à une machine [, qu'il] n'existe aucune proposition sérieuse pour un schéma qui rendrait une machine capable de réaliser des inférences dans des circonstances semblables à celles où un être humain intelligent pourrait les faire.*

Cependant, la prise en compte des aspects sémantiques plus poussés et des aspects pragmatiques de la langue n'est pas totalement ignorée par l'informatique. Elle est étudiée et mise en pratique dans des petits projets expérimentaux. Ces projets sont en fait le peu de

projets de TA qui entrent réellement dans le domaine de l'IA. Ils ne sont pas représentatifs de l'état actuel de la TA mais nous nous devons d'en parler car ils présentent des possibilités non négligeables. Le schéma général de ceux-ci a été donné au point 3.3 de ce chapitre. Dans ces systèmes, que nous avons qualifiés d'*interlingua* sémantique, l'accent est mis sur la base de connaissances (BC) attachées au système et sur les mécanismes d'inférence qu'on lui applique. Les modèles de représentation utilisés dans ces BC sont des modèles conceptuels et non plus des modèles linguistiques.

Pour conclure sur les liens entre TA et IA, nous pouvons dire que s'il est possible que la TA se rapproche un jour de la FAHQT, ce sera avec l'aide des techniques que lui fournit l'IA. Sans entrer dans les détails, parmi ces techniques, nous pouvons en citer les principales : au niveau des représentations de la sémantique, il y a les logiques formelles (prédicats du premier ordre, logiques temporelles, modales, ...), les réseaux sémantiques et les rôles conceptuels. Au niveau de la représentation du contexte, les travaux les plus marquants furent les *frames* ([MINSKY]) et les *scripts* ([SCHANK]) qui sont tous deux des manières de représenter des situations stéréotypées qui peuvent être d'une grande utilité pour la compréhension d'informations non explicitées dans le texte mais fréquentes dans le contexte de celui-ci.

Deuxième partie :

Le système Logos

CHAPITRE III : Description du système Logos

Dans ce chapitre, nous allons décrire l'ensemble du système Logos. Après un bref historique (section 1) et une présentation générale de l'architecture (section 2), nous développerons les enjeux de tous les modules qui composent le système (section 3). Ensuite, nous présenterons les outils mis à la disposition des clients (section 4) ainsi que les améliorations futures qui sont prévues afin d'augmenter la qualité du système (section 5).

Les documents [SCOTT_a], [SCOTT_b] et [SCOTT_c] décrivent de manière générale le système Logos. Bien que nous nous soyons en partie basé sur ces documents pour rédiger ce chapitre, la plupart des informations proviennent des documents internes à la compagnie Logos et des nombreuses séances d'explication qui nous ont été accordées par plusieurs membres de la compagnie.

1. BREF HISTORIQUE.

L'origine du système Logos se situe dans le domaine militaire. En effet, lors de la guerre du Vietnam, le département de la défense américain (DOD) était évidemment très intéressé par un traducteur anglais-vietnamien afin de rendre accessible aux alliés vietnamiens les manuels d'utilisation et autres documents.

A la fin des années soixante, une équipe de moins de 10 personnes a été constituée par monsieur Bud Scott dans le but de créer le prototype d'un système de traduction automatisée fonctionnant sur IBM. En 1970, ce projet a été approuvé par l'armée américaine. Ce système sera utilisé jusqu'en 1973 et est donc reconnu comme étant un système de traduction automatisée fonctionnant à grande échelle, bien que la qualité du résultat était relative.

De 1972 à 1979, plusieurs prototypes ont été créés (anglais-russe, anglais-français, anglais-farsi) afin d'identifier les problèmes et d'étudier les solutions. En 1980, Logos a commencé le développement d'un système dont la langue source était l'allemand. La première langue cible est l'anglais suivit du français et de l'italien. En 1984, Les systèmes ayant l'anglais pour langue

source commenceront à être mis au point. Les langues cibles développées sont, dans l'ordre, le français, l'espagnol, l'allemand et l'italien.

Toutes ces paires de langues ont été développées sur la demande de clients, le système Logos est donc un produit qui est effectivement utilisé par des entreprises privées. C'est pour cela qu'il a été nécessaire de créer des outils (Alex, Semantha) permettant aux utilisateurs d'adapter le produit à leurs besoins.

Les deux dernières années ont surtout vu le changement du système d'exploitation (Wang Word Processor étant remplacé par UNIX) plutôt que le développement de nouvelles paires de langue mais l'emploi de français comme langue source ne devrait plus tarder.

Les projets à l'étude actuellement sont l'utilisation d'une base de donnée relationnelle et la mise en place d'une mémoire de traduction mais ce point sera développé dans la section 5.1.

2. *L'ARCHITECTURE DU SYSTEME*

L'architecture du système Logos peut être vue comme un « pipeline » car réaliser une traduction consiste à appliquer aux phrases une suite de modules sans qu'il y ait de retour à un module précédent. La figure 3.1. montre les différents modules qui permettent de passer d'un document écrit en langage source à un document de même format mais écrit dans un langage cible. Ce schéma différencie les modules qui agissent sur la source (sur fond clair) et ceux qui travaillent sur le résultat (sur fond grisé).

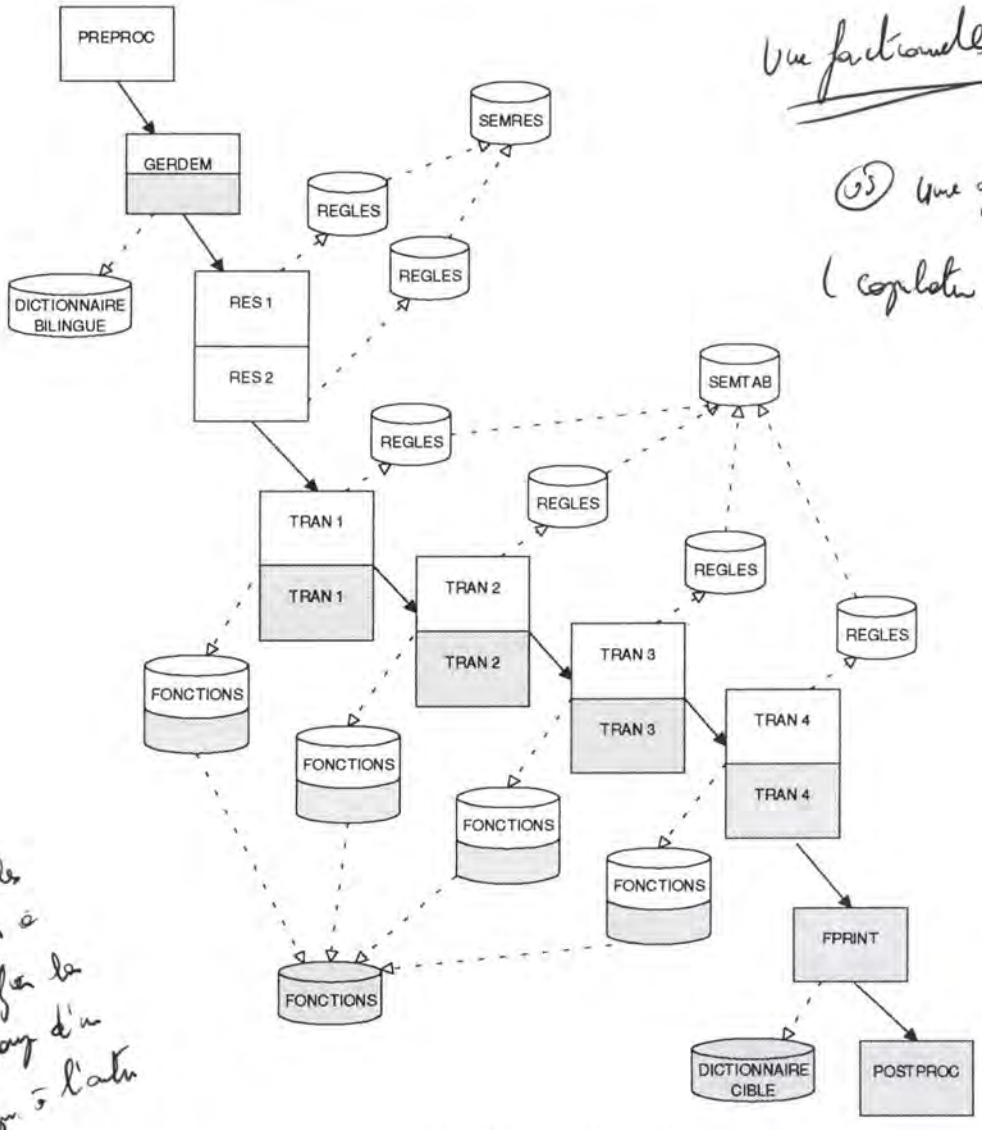


Figure 3.1. : L'architecture du système

Le document source peut provenir de différents traitements de texte et peut contenir des graphiques. Le module Preproc aura donc pour tâche d'extraire le texte à traduire du document et de garder les informations relatives au format. Ces informations seront utilisées par le module Postproc afin que le document final ait la même mise en page que celui d'origine. Certaines informations concernant le format des caractères et qui peuvent être utiles du point de vue linguistique (majuscule, italique, ...) seront également transférées aux modules suivants.

Le module Gerdem reçoit donc une suite de caractères mais aussi quelques informations concernant le format qui peuvent être intéressantes pour la traduction ainsi que des informations quant à la séparation des phrases. Ce module a pour rôles d'identifier les différents éléments de la phrase, d'accéder au dictionnaire et de déterminer définitivement les débuts et fins de phrases. Dans le dictionnaire, on trouve un maximum de 3 rôles syntaxiques

qu'un élément peut jouer dans une phrase plus des caractéristiques relatives à la syntaxe, à la sémantique et à la morphologie de chacune des possibilités. Toutes ces caractéristiques sont représentées par des nombres (*SAL code*), les modules suivants ne travaillerons plus que sur cette représentation. En plus de ces informations, Gerdem donne la traduction par défaut de chaque élément. Les informations contenues dans le dictionnaire sont très rudimentaires car, en fait, la fonction de dictionnaire est répartie dans plusieurs sections du système. Dans la suite du traitement, le texte sera analysé phrase par phrase à l'intérieur de chaque module.

Les modules *Res* et *Trans* fonctionnent selon le même principe qui consiste à appliquer une suite de règles provenant de la base de donnée. Chaque règle dispose d'un pattern et de conditions permettant de déterminer les circonstances nécessaires à l'application de la règle. Ce pattern étant constitué de *SAL codes*, tout comme la représentation interne de la phrase, le choix de la règle (principe du *pattern matching*) à appliquer est très aisé. Comme nous le verrons plus loin, cette particularité permet également de faire croître de manière importante le nombre de règles sans avoir d'impact majeur sur les performances.

Le module *Res* a deux fonctions qui ne concernent que le texte source. La première est de déterminer le rôle syntaxique joué par l'élément parmi les 3 possibilités données par le module précédent. La deuxième est de repérer les différentes relatives ainsi que les propositions. Pour effectuer ces 2 tâches, *Res* effectue 2 passes qui consistent à appliquer un ensemble distinct de règles. La première analyse ayant pour but principal de mettre à la disposition de *Res* 2 un maximum d'informations afin que celui-ci prenne les bonnes décisions. Les règles de chacune de ces passes peuvent accéder à une base de données commune (SEMRES) qui contient des règles complémentaires concernant la sémantique et la morphologie. Lorsque ce module est terminé, une et une seule catégorie syntaxique est associée à chaque élément des phrases et elle ne seront modifiées par la suite que dans de rares occasions par les *Trans*.

Les 4 *Trans* peuvent être considérés comme une analyse réalisée en 4 passes. Chacune ayant pour but d'utiliser les informations dont elle dispose pour résoudre un certain type de problèmes et de mettre un maximum d'informations à la disposition de la passe suivante afin qu'elle puisse résoudre des ambiguïtés d'un niveau supérieur. Comme pour le module *Res*, les différents ensembles de règles auxquels accèdent les *Trans* peuvent faire appel à des règles complémentaires qui leurs sont communes et qui ont principalement trait à la sémantique (SEMTAB). Les fonctions exécutées par ces modules peuvent avoir des conséquences sur le texte cible bien sûr mais aussi sur le texte source (simplification du texte source afin de faciliter l'écriture des règles utilisées par le niveau suivant). Ces fonctions sont écrites dans un langage propre à Logos appelé LPL (*Linguistic Programming Langage*).

Fprint effectue l'opération inverse de Gerdem c'est-à-dire qu'il accède au dictionnaire cible sur base des codes issus des modules de traduction et il génère les formes fléchies. Il a également un rôle linguistique car il s'occupe d'effectuer des élisions, des contractions, ...

Postproc utilise les informations données par Preproc pour créer un document de sortie ayant la même mise en page et le même format que le document source.

Grâce à cette architecture, lors de l'addition d'une paire de langue, il suffit (à quelques exceptions près) d'écrire de nouvelles règles adaptées aux langues concernées. Le système Logos en lui-même (les rectangles sur la figure numéro 3.1.) ne doit donc pratiquement pas être modifié.

3. *LE FONCTIONNEMENT DU SYSTEME.*

3.1. Notions de base

Comme nous venons de le voir, du module Gerdem au module Fprint, le système travaille uniquement sur une représentation numérique des phrases à traduire. Cette représentation est basée sur un ensemble structuré de codes numériques attachés à chaque élément de la phrase. Ces codes sont composés de 3 éléments : la *word class*, le *SAL code* (*type*) et la *form*. La combinaison de ces nombres, augmentée de 4 chiffres supplémentaires appelés *overflows*, permet de faire référence à toutes les informations morphologiques, syntaxiques et sémantiques de la phrase.

3.1.1. Word class

Au départ, le terme *word class* est pour Logos synonyme de *Part-Of-Speech*. Il désigne la classe syntaxique à laquelle appartient un mot donné. Mais, comme nous allons le voir, la notion de classe syntaxique a été un peu détournée de son acception commune en raison des besoins du systèmes.

Les *word classes*, présentées dans le tableau 3.1. sont au nombre de 16 et sont identifiées par un nombre à deux chiffres.

Numéro	Intitulé	Exemples
01	Nouns	Table, man, freedom...
02	Verbs	Work, fly...
03	Locative Adverbs	Here, now, inside...
04	Adjectives	Nice, big...
05	Pronouns	You, myself, few...
06	Adverbs	
11	Logical prepositions	For, with, in terms of...
12	Auxiliaries & modals	Be, can, have to...
13	Locative prepositions	In, where, to a minimum of...
14	Definite determiners	The, this, where...
15	Indefinite determiners	A, more...
16	Arithmates	Two, six...
17	*Not*	
18	Relative or interrogative pronouns	Who, which, how far...
19	Conjunction	And, so that, while, in spite of the fact that...
20	Punctuation	, . ? ...

Tableau 3.1. : Les *word class*

La première remarque que l'on peut faire est que les classes des pronoms, adjectifs, articles et adverbes ont déjà été subdivisées à ce niveau. On a, par exemple, pour les adverbes, décidé de mettre dans la *word class* 03 les adverbes locatifs et de mettre les autres dans la *word class* 06. Cette manière de faire est à attribuer au fait que l'on a voulu regrouper les mots (et groupes de mots) en types qui donnent lieu à des traitements semblables lors du processus de traduction.

Une autre constatation est que les *word classes* 07 à 10 ne sont apparemment pas utilisées. Nous verrons qu'elles ont en fait un rôle de nature différente pour les règles du module *Res*.

Le fait que l'analyse de la phrase tienne compte de la ponctuation dans la structure d'une phrase rend nécessaire l'existence d'une *word class* pour les signes de ponctuation : il s'agit de la *word class* n° 20.

En plus des *word classes* indiquées ci-dessus, il existe aussi ce que l'on appelle des *Super word class*. Elles sont identifiées par des nombres à deux chiffres précédés du signe '-' (on les appelle également *Negative word classes*) et servent à désigner des ensembles de mots (ou groupes de mots) résultant de combinaisons de *word classes*. La plus utilisée est la -01 qui

désigne l'union de toutes les *word classes*. Des combinaisons plus élaborées existent et, dans *Tran* par exemple, -07 est utilisé pour désigner un mot (ou groupe de mots) qui peut être de *word class* 14, 15 ou 16.

Notons enfin que les significations des *word classes* peuvent varier en fonction des besoins des différents modules (*Res*, *Trans* ou *Semtab*). Ceci se fait pas le biais de "normalisations" opérées au début de l'exécution du module. Celles-ci ont pour but de cacher aux règles du module une partie de la complexité des entrées dont on estime qu'il n'a pas besoin pour les traitements qu'il a à faire. Par exemple, au début de *Res*, sous le numéro de *word class* 14, sont regroupées en réalité les *word class* 14 et 15 car la distinction entre déterminant défini et indéfini n'est pas nécessaire pour l'analyse réalisée par ce module.

3.1.2. SAL (Semanto-syntactic Abstraction Language)

SAL est une taxonomie des mots (et groupes de mots) de la langue naturelle à travers trois niveaux d'abstraction. Cette taxonomie permet de gérer une partie de la complexité liée à la richesse sémantique du langage naturel en créant un lien entre syntaxe et sémantique.

Prenons l'exemple de la *word class* 01 contenant les noms. Elle est d'abord divisée en une série de classes appelées *Supersets* telles que « general abstract concepts », « mass nouns », « concrete nouns», ... Comme le montre la figure 3.2., le *Superset* n° 02 « Aspectives » est composé des *Sets* « bearing surface », « functional aspects », « member/portion/part », ... Et le *Set* n° 92 « aggregates » se compose des *Subsets* « numéric aspects », ... Le *Subset* n° 101 « numeric aspects » contient des noms comme « tens », « twentles », « hundreds », ...

Il est évident qu'il existe une taxonomie différente pour chaque *word class*. C'est pour cette raison que certains considèrent que la *SAL code* d'un élément est la combinaison de la *word class* et du *type* formé des 3 nombres représentant chacun un niveau d'abstraction. Par exemple, le nom « table » à comme *SAL code* : 01 03 34 722.

A chaque *Superset*, *Set* ou *Subset* est associé un nombre compris dans un intervalle déterminé.

- *Superset* : 01 à 16
- *Set* : 17 à 99
- *Subset* : 100 à 998

L'attribution de fourchettes différentes pour chaque niveau d'abstraction permet de définir le *type* d'un élément grâce à un seul nombre compris entre 01 et 998. Cette particularité est utilisée pour l'écriture des règles, comme nous le verrons au point II.4.5. lorsque nous décrirons la méthode du *pattern matching*.

Le *SAL code* est un aspect extrêmement important du système. Il permet de traiter les aspects syntaxiques et sémantiques en même temps, chose qui s'est depuis longtemps révélée nécessaire afin de produire des traductions de qualité.

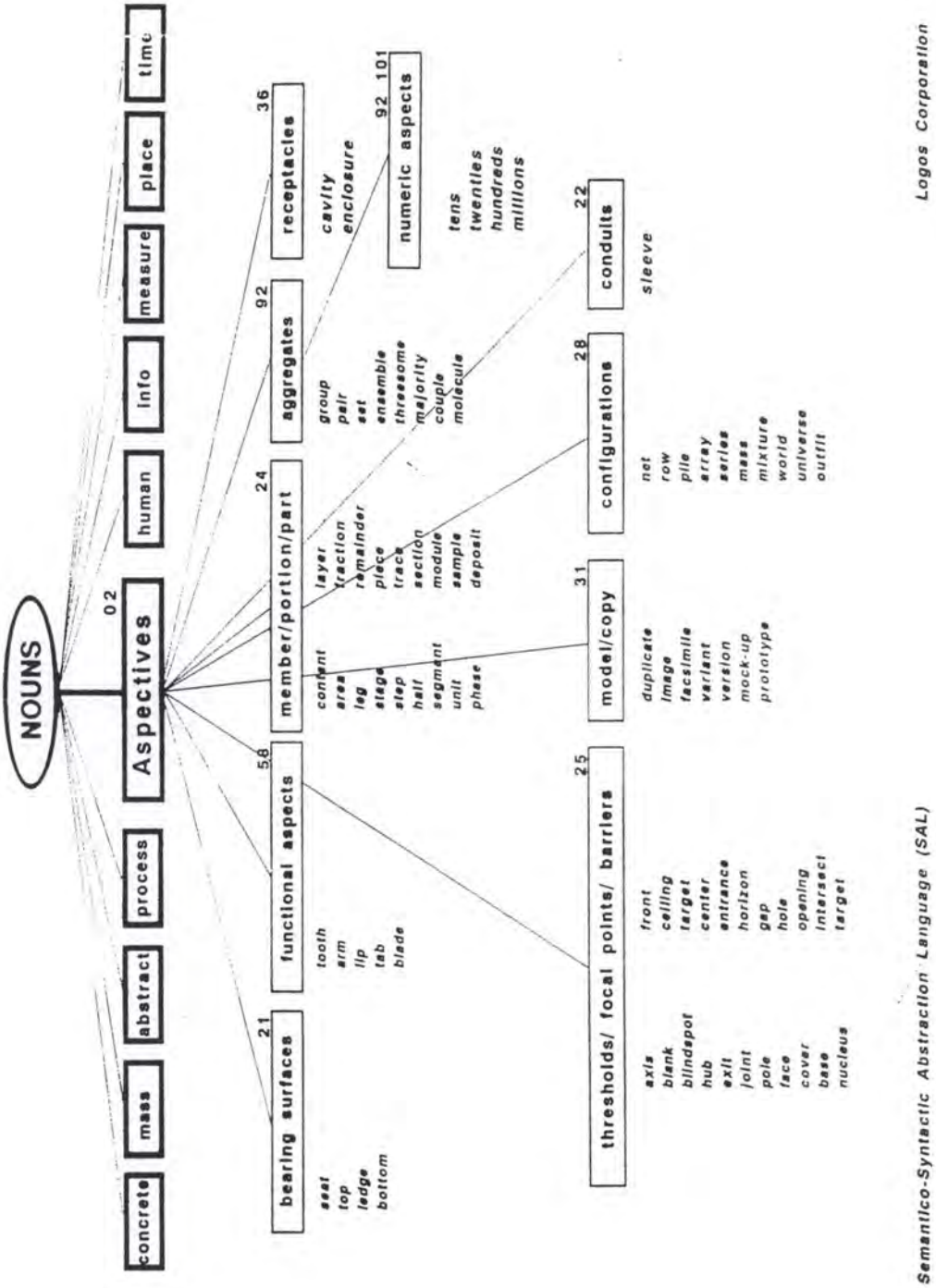


Figure 3.2. : Exemple de Taxonomie *SAL* concernant les noms (*wc*=01)

3.1.3. Les forms

Les *forms* sont des nombres inférieurs à 99 qui ont pour rôle de faire référence aux caractéristiques morphologiques et syntaxiques de l'élément concerné. Tout comme pour les *SAL Code*, la signification des *forms* diffère selon la *word class* concernée.

Exemple

wc = 01	*form* = 01	: nom singulier
wc = 01	*form* = 10	: nom singulier ou pluriel
wc = 02	*form* = 01	: verbe à l'infinitif
wc = 02	*form* = 07	: participe passé

Comme dans le cas des *word classes*, il existe des *forms* spéciales appelées *superforms* qui représentent un ensemble de *forms*. Elles s'échelonnent de 20 à 99 sauf la *superform* universelle qui est égale à -01.

3.1.4. Les overflows

Tous les éléments disposent de 4 *overflows* formés d'un chiffre chacun qui sont associés à diverses informations de types sémantico-syntaxiques et/ou morphologiques. Ces informations ne concernent pas seulement le mot dans la langue source mais peut également concerner l'équivalent dans la langue cible.

3.2. Les dictionnaires

Rappelons d'abord qu'en TA un dictionnaire (ou lexique) est la partie du système dans laquelle sont rassemblées toutes les informations pouvant être associées à des mots : les informations lexicales. Ces informations sont d'ordre morphologique, syntaxique et sémantique. Nous les détaillerons en observant une entrée d'un des dictionnaires du système. Mais voyons d'abord comment sont organisés les dictionnaires.

3.2.1. Organisation des dictionnaires

Un première distinction doit être faite entre *language-pair dictionaries* et *constant dictionaries*.

A) Les language-pair dictionaries

Comme son nom l'indique, un *language-pair dictionary* existe pour chaque paire de langues dans laquelle le système effectue des traductions. Celui-ci regroupe tant les données relatives à l'entrée dans la langue source que les données relatives à sa traduction par défaut dans la langue source. Les *language-pair dictionaries* sont donc des dictionnaires bilingues. Au niveau physique, on a séparé en plusieurs fichiers les données de la langue source et de la langue cible et on parle parfois de *source dictionary* pour designer l'ensemble des fichiers d'un *language-pair dictionary* qui contiennent les données de la langue source et de *target dictionary* pour ceux qui contiennent les données de la langue cible. Chaque entrée du *source dictionary* contient les pointeurs vers sa traduction par défaut qui se trouve dans le *target dictionary*.

Le *language-pair dictionary* lié à la paire de langues dans laquelle on veut traduire est consulté au début de la traduction par le module *Gerdem* qui associe à chaque élément de la phrase correspondant à une entrée du *language-pair dictionary* une partie des données qui l'accompagnent. Celles-ci seront détaillées dans le point suivant.

Nous n'entrerons pas ici dans les détails de la répartition des dictionnaires en divers fichiers car celle-ci relève d'une logique d'optimisation des accès liée à l'implémentation en FORTRAN qu'il n'est pas utile d'exposer dans le présent travail. Signalons seulement qu'actuellement un petit nombre d'entrées les plus fréquemment utilisées de chaque *language-pair dictionary* est placé en mémoire centrale plutôt que sur disque. On a donc une répartition en deux sous-dictionnaires : l'un, l'*internal dictionary*, est tenu en mémoire centrale et contient les entrées les plus fréquemment utilisées tandis que l'autre, l'*external dictionary*, se trouve sur disque et regroupe toutes les autres entrées, c'est-à-dire la majorité d'entre elles. Lorsque le module *Gerdem* essaie d'établir une correspondance entre les éléments de la phrase à traduire et les entrées du dictionnaire, il consulte d'abord l'*internal dictionary* pour des raisons évidentes de gain de temps.

B) Les constant dictionaries

Les *constant dictionaries* sont relatifs à une langue cible et ne contiennent que des informations sur les mots de la langue cible. Les *constant dictionaries* sont donc des dictionnaires monolingues et sont utilisés par le module *Fprint* pour générer la suite de caractères qui correspond au code fourni par les règles des modules *Trans*. En effet, ces règles ont la possibilité de modifier la traduction de mots en donnant l'adresse de la nouvelle traduction dans le *constant dictionary* si certaines conditions sur le contexte syntaxique et

sémantique sont rencontrées. Une différence entre ces traductions et celles qui sont contenues dans les *target dictionaries* est que leur adresse est contenue dans des règles et non dans les entrées d'un *source dictionary*. Le nom de *constant dictionaries* provient du fait que chaque mot de la langue cible qu'ils contiennent est désigné par une valeur numérique constante.

Les *constant dictionaries* sont, tout comme les *language-pair dictionaries*, répartis en deux sous-dictionnaires. Le *High Frequency Constant Dictionary* est utilisé par les règles de *Tran* alors que le *Low Frequency Constant Dictionary* est consulté lors des appels à *Semtab*.

3.2.2. Language-pair dictionary ou constant dictionary

Le nombre de traductions pour une même entrée a été limité à trois dans les *language-pair dictionaries*. Dans les cas où un mot (ou, moins vraisemblablement, un groupe de mots) admet plus d'une traduction pour une même *part-of-speech*, les autres traductions sont prises en charge par *Tran* via *Semtab* et le *constant dictionary*. On se pose alors naturellement la question de savoir quelles traductions on va mettre dans quel dictionnaire. La stratégie de départ de Logos suivait le principe selon lequel la traduction qui est la plus difficile à déterminer doit se trouver dans le *language-pair dictionnary* afin de constituer la traduction par défaut. Cependant, la pratique a montré que le fait de placer la traduction la plus courante comme traduction par défaut s'avérait plus payante dans certains cas.

3.2.3. Qu'est-ce qui constitue une entrée ?

L'objectif de la compagnie Logos ayant toujours été de créer un système *general purpose*, la quantité de mots qu'il peut être amené à rencontrer est très importante. Disposer de l'ensemble des mots, sous toutes leurs formes, comme entrée du dictionnaire entraînerait donc un volume très important de données à gérer. Pour diminuer ce volume, la stratégie adéquate consiste à ne considérer que les formes canoniques comme entrées à part entière. Cependant, la gestion des exceptions est alors très complexes. Finalement, la stratégie choisie par Logos a été de créer une entrée pour les racines et une pour les formes fléchies qui constituent une exception.

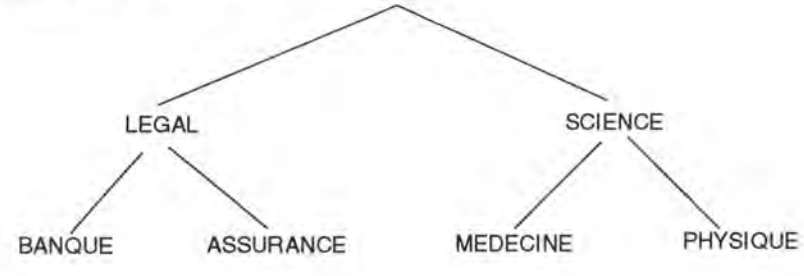
Exemple

L'entrée correspondant au mot « works » sera la même que pour le mot « work » puisque celui-ci représente en même temps la forme canonique de « works ». Par contre, il y aura 2 entrées distinctes pour les mots « foot » et « feet ».

Le dictionnaire accepte comme entrée aussi bien des mots que des groupes de mots. Il n'en a pas toujours été de la sorte puisque avant seuls les mots étaient autorisés. On a laissé tomber cette limitation et considéré qu'un groupe de mots devait constituer une entrée du dictionnaire si « sa signification est plus que la somme de ses parties » et si « son utilisation est extrêmement commune ». Ce mécanisme a pour but d'éviter que des expressions comme « heavy duty truck » ne soient traduites par quelque chose comme « un camion avec de grandes responsabilités ». Nous verrons cependant au chapitre IV réservée à la critique que ce choix peut être sujet à caution.

Un des problèmes rencontrés par les systèmes *general purpose* concerne la sélection de la bonne traduction suivant le contexte sémantique. Il s'agit évidemment d'une des tâches naturellement attribuées au processus même de traduction mais cela constitue une problématique très importante pour la Traduction Automatique en général. Afin de faciliter ce choix, le système Logos dispose de codes fournis par l'utilisateur avant le lancement de la traduction. Ces codes se trouvent sous forme de hiérarchie et sont de 2 types : les *subject matter code* et les *company codes*. Les *subject matter code* font référence au contexte, ils indiquent qu'il s'agit d'un texte ayant trait à la médecine, à l'informatique, ... Quant aux *company codes*, il permettent aux différents clients de personnaliser le système selon leurs besoins spécifiques.

Exemple de hiérarchie de *subject matter code*



Lorsqu'un linguiste désire insérer une entrée selon un code particulier, il s'agit la plupart du temps d'une seule *part-of-speech*. Mais il est possible que pour l'entrée par défaut, il existe d'autres *parts-of-speech*. C'est pour empêcher que des erreurs soient commises en raison de la sélection d'un code qu'un système de migration des *part-of-speech* a été mis au point. Le principe étant que si un linguiste insère une nouvelle entrée correspondant à un *SMC* ou un *CC* et qu'il existe déjà une entrée par défaut pour cet élément, les *part-of-speech* de cette entrée seront automatiquement ajoutées à celle qui a été donnée par le linguiste.

3.2.4. Le contenu des dictionnaires

Ci-dessous, nous présentons et commentons une entrée de l'*external dictionary* dans le but de montrer quelles sont les connaissances lexicales contenues dans le système.

Exemple d'entrée de l'*external dictionnary*

Voici les données se rapportant au mot 'file'. On retrouve ici tant les données du *source dictionary* que celles du *target dictionary* auxquelles il est fait référence :

```
                                        SMC: 01000    OPID: 1    HASH: 25032006
FILE$

        CODE : 1  76  1  76  01  01  12              SRCPAT :  1  1  TRANADR:  68834
       TRANS : FICHIER$                             GENDER :  N/A
       STEM  : FICHIER, FICHIER                     TGTPAT : 1  1  120  111

        CODE : 2  24  1  212 05  04  10              SRCPAT :  2  1  TRANADR:  68826
       TRANS : INSCRIRE$                            GENDER :  N/A
        STEM : INSCRI,INSCRIV,INSCRIVI,INSCRIVI3,   TGTPAT : 5  2  62  119
               INSCRIT,INSCRIPTION

        CODE : 2  68  1  212  08  41  7             SRCPAT :  2  1  TRANADR:  68827
       TRANS : CLASSER$                             GENDER :  N/A
        STEM : CLASS,CLASSA3,CLASSE,CLASSE1,CLASSER TGTPAT : 4  1  3  120

                                        SMC: 31224    OPID: 1    HASH: 25032006
FILE$

        CODE : 1  76  1  76  01  01  12              SRCPAT :  1  1  TRANADR:  50710
       TRANS : FICHIER$                             GENDER :  N/A
       STEM  : FICHIER, FICHIER                     TGTPAT : 1  1  120  111

        CODE : 2  24  1  212 05  04  10              SRCPAT :  2  1  TRANADR:  17085
       TRANS : INSCRIRE$                            GENDER :  N/A
        STEM : INSCRI, INSCRIV,INSCRIVI,INSCRIVI3,  TGTPAT : 5  2  62  119
               INSCRIT,INSCRIPTION

        CODE : 2  68  1  212  08  41  7             SRCPAT :  2  1  TRANADR:  50711
       TRANS : CLASSER$                             GENDER :  N/A
        STEM : CLASS,CLASSA3,CLASSE,CLASSE1,        TGTPAT : 4  1  3  120
               CLASSEMENT
```

Le *matching* se fait sur la forme canonique. Donc, si dans la phrase source on trouve le mot 'files', on n'établira une première correspondance que sur les quatre premières lettres de celui-ci. Les principes selon lesquels se fait le *pattern matching* seront expliqués plus longuement lors de la présentation du module *Gerdem* mais signalons seulement que c'est le symbole '$' suivant la chaîne de caractères 'FILE' qui indique que le mot peut être fléchi. Il faut alors de vérifier que la chaîne 'S' fait partie des terminaisons possibles du mot.

On constate que deux entrées ont été trouvées. Elles correspondent à des *subject matter codes (SMC)* différents. Pour tout mot, il y a toujours au moins une entrée correspondant à la traduction par défaut (*SMC*=01000 appelé *general entry*) c-à-d l'emploi des mots dans un contexte général. La seconde entrée dispose d'un *SMC* égal à 31224, c-à-d la catégorie 'Information processing'. Si l'utilisateur sélectionne le code 31224 avant la traduction pour indiquer que le texte fait référence au domaine de traitement de l'information, cette seconde entrée sera prise en compte à la place de l'entrée par défaut.

OPID désigne le numéro de la personne qui a encodé l'entrée.

HASH (*Hashcode*) est un nombre qui doit jouer un rôle d'identifiant pour les entrées du dictionnaire. Il ne s'agit pas réellement d'un identifiant car les entrées composées de plusieurs mots reçoivent le même *Hashcode* que le nom qui est considéré comme l'élément principal de l'expression. Malgré cela, il est possible de faire référence à une seule et unique entrée du dictionnaire par l'intermédiaire d'un autre code appelé *hennum*.

Ensuite, on trouve, pour chaque entrée, la liste de ses traductions possibles. Leur nombre est limité à 3 et il doit s'agir de *word classes* différentes (sauf pour la *word class* 2 - les verbes - pour laquelle on peut avoir deux traductions à condition que l'une soit un verbe transitif et l'autre un verbe intransitif). Cette limitation facilite l'étape de résolution des homographes (effectuée par le module *Res*) car les règles doivent tenir compte de beaucoup moins de possibilités quant aux suites possibles d'éléments qu'il lui faut analyser. Evidemment, la traduction peut de ce fait être erronée et ce sont les modules *Trans*, via *Semtab*, qui seront chargés de pallier les déficiences de cette première 'désambiguätion' en faisant référence à d'autres traductions qui se trouvent dans les *constant dictionaries*. Mais notons également qu'une telle manière de faire est possible car le taux d'erreur concédé par *Res* est très faible.

Le *CODE* donne, dans l'ordre, les valeurs des *word class, Set, Form, Subset (FOL 1), FOL 2A, FOL 2B, FOL 3A, FOL 3B, Superset (OFL4)*. En observant la *word class* pour chacune des 3 traductions de la première entrée de l'exemple, on voit qu'il y a un nom et deux verbes, l'un intransitif et l'autre transitif.

Les deux nombres suivant *SRCPAT* renvoient respectivement à la *PAT-table* (ensemble de données morphologiques sur le mot en langue source) et à la *Stem* (racine) de ce mot. Dans l'exemple, le numéro de *PAT-table* change selon qu'il s'agit d'un nom ou d'un verbe. C'est normal vu que les terminaisons autorisées ne sont pas les mêmes selon le cas. Par contre, le numéro de *Stem* est toujours le même. Il en est ainsi la plupart du temps sauf dans le cas de mots ayant des inflexions irrégulières comme 'mouse' qui au pluriel donne 'mice'. En effet, dans ce cas, il existe une entrée particulière pour 'mice' et on signale dans la valeur de *Stem* (=2) qu'il ne s'agit pas de la première racine du mot.

TRANADR, abréviation de 'transfer address', indique l'adresse du *target dictionary* à laquelle se trouve telle traduction du mot source. Quand *Gerdem* associe à chaque mot de la phrase, les informations qu'il trouve dans le dictionnaire, il ne charge en mémoire centrale que celles qui sont nécessaires à l'analyse c'est-à-dire les 7 nombres qui suivent le mot *Code*. Les informations qui suivent (à part *GENDER*) ne sont nécessaires que pour la génération et ne

seront retrouvées dans le *target dictionary* qu'une fois l'analyse terminée. C'est le module *Fprint* qui s'en chargera et ce grâce à l'adresse de traduction qui lui sera communiquée par *Tran4*.

GENDER indique le genre du mot source. Il n'est indiqué que lorsque la langue source est l'allemand.

TRANS donne la forme canonique de la traduction. Ici aussi le signe '$' signifie que le mot peut avoir des formes fléchies. Cependant, contrairement au mot dans la langue source sur lequel est effectué le *matching*, ici les terminaisons possibles du mots ne viendront pas se greffer à la fin de sa forme canonique mais à la fin de ses différentes racines. Ces racines sont données à la suite du mot *STEM* tant pour la *main word class* que pour l'*alternate word class* ce qui permet également de surmonter les restrictions portant sur le nombre de traductions.

Enfin, *TGTPAT* indique les données liées aux *PAT-table* de la traduction. Le premier nombre est le nombre de *stem* du transfert. Le deuxième représente le genre du nom (1 = masculin, 2 = féminin, 3 = neutre) si la traduction ou son *alternate word class* est un nom. Les deux derniers nombres indiquent le numéro de *PAT-table* correspondant respectivement à la traduction et à son *alternate word class*.

3.3. Le module Preproc

Le module *Preproc* (abréviation de « Preprocessing ») est la première étape du processus même si elle ne fait pas à proprement parler partie de la traduction. Comme nous allons le voir, *Preproc* agit de manière plutôt mécanique et la quantité d'information linguistique qu'il contient est limitée.

Le but de *Preproc* est double. D'une part, pour les besoins de la traduction, il doit mettre le texte sous une forme conventionnelle. D'autre part, comme les documents à traduire peuvent provenir de divers logiciels de traitement de texte et de desktop publishing, il est nécessaire de séparer les instructions de formatages du texte lui-même.

Preproc reçoit en entrée un texte sous un des formats suivants :

- troff,
- scripta,
- FrameMaker MIF (Maker Interchange Format),

- Microsoft Word RTF (Rich Text Format),
- WordPerfect,
- Interleaf ASCII,
- SGML (Standard Generalized Markup Language).

En sortie, *Preproc* produit deux fichiers :

- un fichier LTX (Logos Text File)[4],
- un fichier contenant des instructions de formatage (Logos Markup File).

Le fichier LTX contient le texte sous forme normalisée et c'est ce qui devrait être passé au module *Gerdem* afin de véritablement commencer la traduction. Cependant, pour l'instant, le reste du programme n'ayant pas encore été modifié pour accepter ce format, le fichier LTX est transformé en fichier *LTEL* (ancien format accepté par le reste du programme).

Le fichier contenant les instructions de formatage sera utilisé, une fois la traduction effectuée, par le module *Postproc* dont le but est de fournir à l'utilisateur un document cible ayant la même présentation que le document source.

Attardons-nous maintenant aux principales opérations que doit effectuer *Preproc*.

3.3.1. La normalisation du texte

Nous allons voir cela aux trois niveaux qu'il est utile de distinguer pour cette tâche : les caractères, les mots et les phrases.

A) Au niveau des caractères

Les caractères sont convertis en format LTX par une sous-routine propre au type du document source.

B) Au niveau des mots

[4] LTX est un nouveau format de texte adopté par Logos et qui est basé sur la norme ISO 8859/1 (Latin-1). Ses principaux avantages par rapport à ltel (Logos Telegraphic Code, ancien format) sont l'utilisation d'accents et la présence de caratères de contrôle affichables : le tout fait qu'il est plus lisible et que sa conversion avec d'autres formats est plus aisée.

La fin d'un mot est établie si l'on se trouve dans l'un des cas suivants :

- quand on rencontre un espace ou d'autres caractères tels que : , [] () ; etc.,
- quand le mot est trop long pour être mis dans le buffer,
- quand on rencontre certaines instructions de formatage propres à tel ou tel type de document comme les espaces insécables en format RTF.

Preproc s'occupe également de remplacer les contractions par leur forme non contractée. Chaque fois qu'une apostrophe est rencontrée, on fait appel à une routine qui consulte la table des contractions. Celle-ci est lue une fois séquentiellement jusqu'à ce qu'une correspondance soit établie. Pour cette raison, on a classé les cas particuliers avant les cas généraux (CAN'T est avant N'T) et, lorsqu'une même contraction a différentes utilités, seule la plus fréquente est retenue (par exemple, 'D est plus souvent utilisé au lieu de WOULD qu'au lieu de HAD). Notons également que la correspondance n'est pas forcément établie sur des mots entiers mais peut se base sur des partie de ceux-ci. Ainsi, dans la table, l'entrée HE'S sert aussi bien a transformer HE'S en HE IS que SHE'S en SHE IS.

Le traitement des contractions contient donc une « intelligence » linguistique assez réduite et se limite en définitive à des considérations statistiques.

C) Au niveau des phrases

La phrase étant l'« unité » sur laquelle s'effectue le processus de traduction, *Preproc* lit d'abord tous les mots jusqu'à ce qu'il détecte une fin de phrase. Il effectue ensuite les opérations sur les mots et les caractères avant d'écrire la phrase dans le fichier de sortie. L'identification de la fin d'une phrase est cependant un problème plus compliqué qu'il n'y paraît à première vue. *Preproc* considère qu'il a atteint la fin d'une phrase s'il rencontre :

- la fin d'un document,
- un élément « spécial » d'un document tel qu'un tableau, un dessin, ...
- un caractère de fin de phrase (. ! ? :) sauf si le mot qui suit commence par une minuscule ou si le dernier mot de la phrase appartient à la liste des *nervereos* (never end of sentence). Cette liste contient des abréviations fréquemment utilisées et qui sont suivies d'un point comme 'Mr.' ...

A nouveau, les tests qui sont effectués sont assez sommaires et il arrive que des erreurs apparaissent lorsqu'on utilise des abréviations.

Exemple

La phrase : « The U.S. government decided to raise taxes ». Le seul moyen dont dispose le système Logos pour que cette phrase ne soit pas coupée au niveau de « U.S. », est de placer cette expression dans la liste des *nevereos*. Or, il est difficile de faire le recensement de toutes les expressions de ce type et, surtout, le système ne sera plus en mesure d'indentifier la fin d'une phrase se terminant par « U.S. ».

Il est cependant important de noter que les cas où de telles erreurs peuvent se produire sont rares et c'est ce qui justifie qu'un examen plus approfondi de la phrase ne soit pas effectué.

3.3.2. La séparation du texte et des instructions de formatage

Pour ce problème plutôt technique nous ne donnons qu'un aperçu du problème.

La principale difficulté rencontrée par *Preproc* est de garder une correspondance entre les deux fichier créés. En effet, il faut pouvoir dire à quelle partie de texte se rattache telle suite d'instructions de formatage afin que *Postproc* puisse correctement recombiner les deux fichiers après la traduction.

Si le formatage se rapporte à une phrase toute entière, cela ne pose pas de problèmes. Il suffit d'établir un parallélisme entre les deux fichiers. Le problème se complique par contre lorsque des mots isolés sont concernés. Dans ce cas, la correspondance est établie via l'utilisation de séquences d'échappement à l'intérieur du texte (fichier LTX) qui renvoient à des instructions du fichier de formatage.

3.4. Le module Gerdem

La tâche principale de Gerdem est d'accéder au dictionnaire bilingue et de placer les informations obtenues dans un tableau qui sera utilisé pendant toute la suite du processus de traduction. Gerdem reçoit de Preproc le fichier *LTEL* contenant le texte et les séquences d'échappement. La première opération à effectuer consiste à identifier les différents éléments qui se trouvent dans ce fichier afin de consulter le dictionnaire.

3.4.1. Identification des éléments

Nous avons vu que Preproc délimitait déjà les débuts et les fins de mots, ces-derniers constituent évidemment des éléments mais il existe d'autres suites de caractères qui sont considérées comme tel :

- *un signe de ponctuation* : il s'agit ici des signes de ponctuation n'indiquant pas une fin de phrase, c'est-à-dire une virgule, un point virgule ou un tiret.

- *une chaîne alphanumérique* (exemples: « 120 », « $30 », ...)

- *une chaîne à ne pas traduire* : l'utilisateur peut déterminer un mot ou plusieurs mots qui ne doivent pas être traduits par le système. Il lui suffit pour cela de placer les mots concernés entre 2 symboles identiques qu'il aura lui-même choisi auparavant.

- *un signe de ponctuation indiquant la fin de la phrase* : ce signe reçoit le nom de *EOS* (*End Of Sentence*)

De plus, un élément artificiel est systématiquement inséré au début de chaque phrase, il s'agit du *BOS* (*Begin Of Sentence).*

3.4.2. Consultation du dictionnaire

L'opération de sélection de l'entrée qui est considérée comme le meilleur choix peut être divisé en 2 étapes. La première consiste à établir une liste triée contenant toutes les entrées susceptibles de correspondre à l'élément recherché. La seconde a évidemment pour but de faire la sélection au sein de la liste.

Comme nous l'avons vu lors de la description du dictionnaire (voir section 3.2.), un mot ne dispose d'entrées distinctes que dans le cas où ses formes fléchies ne se basent pas sur la même racine. Une étape préalable à l'établissement de la liste consiste donc à vérifier (par l'intermédiaire d'une table des inflexions appelée *PAT table*) si la fin du mot ne représente pas une forme fléchie. De cette manière, le système dispose de toutes les racines qui peuvent correspondre à l'élément recherché. On établit alors la liste sur base des entrées dont le premier élément correspond au mot lui-même ou à une des racines possibles identifiées.

Pour faciliter l'étape de sélection de l'entrée, la liste est triée selon plusieurs critères de manière à obtenir les entrées selon un ordre décroissant de priorité. La première entrée qui corresponde entièrement sera donc automatiquement celle qui représente le meilleur choix. Les critères, classé par ordre décroissant de priorité, sont les suivants :

1. *la longueur de l'entrée* : la stratégie de Logos concernant les entrées comprenant plusieurs mots est que le plus long contient le plus d'information, et donc, constitue le meilleur choix.

2. *les company codes* : l'utilisateur peut en définir 5 auxquels il veut donner la priorité. Entre les différents *company codes* choisis, la priorité est déterminée par l'ordre dans lequel ils sont sélectionnés.

3. *les subject matter codes* : pour un même *company codes*, les entrées correspondant à des *subject matter codes* différents (5 au maximum) sont également triées selon l'ordre prédéfini par l'utilisateur.

La sélection de l'entrée la plus prioritaire se résume à parcourir la liste et à comparer les entrées avec les éléments courants de la phrase. Cette comparaison est effectuée élément par élément à l'intérieur d'une entrée (chacun pouvant faire l'objet d'une analyse morphologique puisque plusieurs éléments d'une entrée peuvent être représentés par leur racine). Si aucune entrée de la liste ne correspond, l'élément de la phrase reçoit un *SAL code* spécial représentant un mot inconnu (il faut remarquer que ce sont les mêmes valeurs qui sont attribuées à un mot à ne pas traduire).

3.4.3. Présentation des données obtenues

Toutes les données reçues du dictionnaire sont placées dans un tableau appelé SWORK. Chaque ligne de ce tableau correspond à un élément identifié dans la phrase. Les mots composés qui ont été trouvés dans le dictionnaire sont considérés désormais comme étant un seul élément).

Exemple

« Yesterday, John drove a %Porsche% at 120 miles per hour »

Eléments de la phrase	Première *pos*	Seconde *pos*	Troisième *pos*
BOS	20 01 01		
YESTERDAY	03 10 91 540 01		
,	20 02 888 02		
JOHN	01 01 33		
DROVE	01 05 51 208 01	02 10 24 890 09	02 10 24 890 09
A	14 10 42 315 01		
PORSCHE	01 01 33		
AT	13 05 75 599 03		
120	16 4 21 567 01	14 09 21 567 1	01 13 21 567 09
MILES PER HOUR	01 08 61 170 01		
EOS	20 10 01		

Tableau 3.2. : Exemple de SWORK à la fin de Gerdem. Ce tableau ne comporte que les codes utilisés pour la sélection des règles.

On constate sur le tableau 3.2. que les éléments « john », qui est inconnu du système, et « Porsche », qu'on ne veut pas traduire diposent du même code.

Les informations concernant le format sont placées dans un tableau que Gerdem met à jour en parallèle avec le tableau SWORK afin de conserver la correspondance entre les séquences d'échappement et les éléments visés. Les informations de ce type qui sont nécessaires au processus de traduction sont mises à la disposition des modules suivant par l'attribution de codes numériques qui sont également placés dans le SWORK).

Actuellement, le nombre d'éléments que peut contenir le tableau SWORK pour une même phase ne peut dépasser 70. Si Gerdem s'aperçoit que cette limite est dépassée, il tente de séparer la phrase à un point-virgule ou à un tiret. S'il n'en trouve pas, il coupe la phrase de manière arbitraire après le $50^{ième}$ élément. Gerdem peut donc revenir sur la découpe en phrase qui a été faite par Preproc.

3.5. Le module RES

3.5.1. But

Le module RES analyse la phrase de gauche à droite et son but est double. Le premier est de déterminer le rôle syntaxique de chaque élément de la phrase (cette phase est particulièrement importante pour l'anglais en raison du grand nombre d'homographe). Pour cela, il identifie parmi les 3 *part-of-speech* possibles qui se trouvent dans le dictionnaire laquelle est utilisée dans la phrase analysée. Si on considère chaque rôle syntaxique comme un noeud, la tâche de RES consiste à déterminer parmi tous les chemins possibles, celui qui correspond à la phrase analysée.

Exemple

« When the light flashes, you may press the button. »

La figure 3.3. indique en gras le chemin identifié par *Res* pour cette phrase.

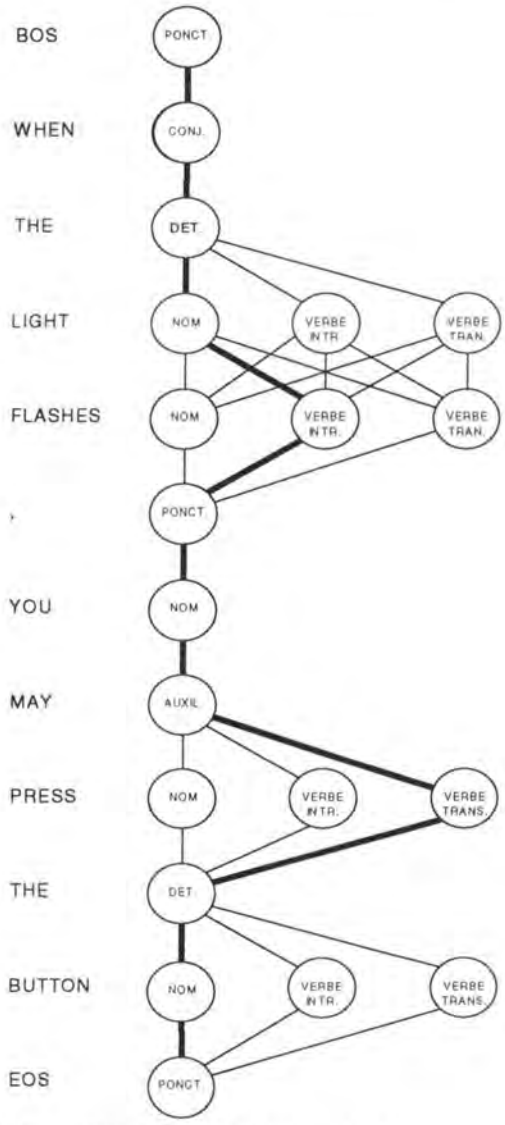


Figure 3.3. : Exemple de sélection des noeuds

A la fin de ce module, un et un seul noeud sera désigné pour chaque élément de la phrase afin qu'il ne reste plus aucune ambiguïté. Cela signifie que dans le cas où aucune règle ne permet pas de déterminer le rôle syntaxique de l'élément, une des 3 possibilités sera choisie par défaut. Nous verrons qu'en fait, c'est la catégorie syntaxique qui est la plus difficile à prouver qui sera sélectionnée dans ce cas.

Le deuxième but du module RES est de délimiter les principaux composants de la phrase analysée tels que les débuts et fins de clauses, de propositions, ... Comme nous le verrons plus loin, cette analyse effectuée au départ d'une manière *bottom-up*, a été transformée (réalisation en 2 passes) pour arriver à une méthode mixte *bottom-up* et *top-down*.

La grande majorité des mots anglais n'ont pas plus de 3 homographes mais certains peuvent avoir jusqu'à 7 *part-of-speech* différentes (telles qu'elles sont définies par Logos). On constate

la première limite de RES. Si le rôle syntaxique d'un élément n'est pas un des 3 qui se trouvent dans le dictionnaire, ce module ne dispose d'aucun moyen pour le savoir. Par contre, les modules suivants (les TRANS) pourront, dans certains cas, corriger ces erreurs grâce aux informations supplémentaires dont ils disposent.

Exemple

« He bought a building. »

Dans cette phrase, *« building »* est un nom concret mais comme il peut avoir plus de 3 autres homographes (forme progressive du verbe, participe présent du verbe transitif, participe présent du verbe intransitif, nom représentant un processus, ...), cette *part-of-speech* ne se trouve pas dans le dictionnaire. On aura donc obligatoirement une erreur au niveau de cet élément à la fin de RES.

L'analyse de la phrase est effectuée en 2 passes (RES1 et RES2). La première ne résout que très peu d'ambiguïtés, son rôle principal étant de mettre un maximum d'informations a la disposition de la deuxième passe.

3.5.2. Les règles

Les 2 phases consistent en l'application d'une suite de règles, une partie de ces règles appartiennent à un ensemble distinct pour chaque passe (environs 3100 règles pour *Res*1 et 9700 pour *Res*2 en ce qui concerne l'anglais) et il existe également une partie commune (SEMRES). *Tran* et *Res* utilisent tout les deux le principe du *pattern matching* et disposent de règles dont la structure est fortement similaire.

Exemple de règle contenant un appel à SEMRES

```
POSSIBLE MATCH GOING TO RESSEM AT 10 RULE NO. 2230 IS
  3     14  -1 -1      1  -1 42     -6**** -1        0    52307
        -22 -81 -82    0    0    0 999
      *** SEMWRK VALUES ***
     14    1  3 10      14    1  3 10       1  34  1 11
       101    1    1       101    1    1       720    3  34
                          *** RES22 MATCH
                    3      14 101  1      10 101 -1      1  -1 42          0 6229
                  999
MATCH AT 10 RULE NO. 2230
3     14  -1 -1      1  -1 42     -6**** -1        0    52307
      6014   81    18 9000 6014   82    12 9000
        -22 -81 -82    0    0    0 999
```

A) Structure des règles

Les règles sont formées de 4 composants qui sont : le commentaire, le pattern, les conditions et le VTR.

Le commentaire est représenté par une ligne écrite suivant une syntaxe définie, il résume brièvement les circonstances dans lesquelles la règle devra être appliquée et quels en sont les effets.

Exemple

```
(V-SIMP PRE-CL)(C7-3)(N U/PV2RT-PL) N(PL) EL(NON-N,WC18)=-1 C7=0 S487
```

Le pattern indique au système les caractéristiques syntaxiques et sémantiques que doivent posséder les éléments analysés pour que la règle puisse être appliquée. Ce pattern est constitué d'une suite d'éléments qui ont chacun un format qui correspond à un SWORK. Cette méthode donne la possibilité de comparer efficacement la partie de phrase analysée avec le pattern. De plus, la présence du type dans tous les éléments du pattern permet de choisir le niveau d'abstraction (*superset*, *set* ou *subset*) qui convient le mieux aux besoins spécifiques de chaque règle.

Exemple

```
20   1  1      2  21 67     14  -1 31
```

Ce pattern est composé de trois éléments qui sont composés chacun d'une *word class*, d'un *type* et d'une *form*.

Le but des conditions est de pallier au manque d'expressivité du pattern qui, bien qu'il permette un accès efficace aux règles, ne dispose pas de la flexibilité qui est nécessaire pour gérer de manière satisfaisante la complexité du langage naturel. Pour cela, le système a mis à la disposition de la personne qui écrit les règles des conditions qui permettent d'augmenter ou de diminuer leur généralité. Pour rendre une règle plus spécifique, il est possible de donner un code qui ne correspond qu'à un seul mot (*hash code*) ou à tous les groupes nominaux qui possèdent le même nom principal (*hennum code*). Par contre, pour rendre une règle plus générale, il est possible de donner une liste de *word class, superset, set* et *subset* auxquels les règles s'appliquent. Si on a besoin d'être très général, on peut même donner uniquement la liste des caractéristiques auxquelles il ne faut pas appliquer la règle.

Exemple

```
6050 1007 8003 1103 8058 1223 8009 9000      6014   82    1   18 9000
```

Toutes les conditions sont composées d'une suite de nombres de 4 chiffres. Elles sont identifiées par un nombre commençant par 6 (il y en a 2 dans l'exemple ci-dessus). Les nombres qui suivent sont les paramètres qui indiquent la condition proprement dite ainsi que l'élément du pattern auquel elle s'applique. C'est la présence du nombre 9000 qui indique la fin d'une condition.

Le dernier élément est le VTR, il s'agit d'une suite de fonctions (appelées *switch*) écrites dans un langage interne au système Logos (*Linguistic Programming Language*) qui sont interprétées et exécutées lorsque la règle est désignée comme celle qu'il faut appliquer.

Exemple

```
-22 -82 -81 -83   0   0 -17 183 -82   1 -18  10  20 -82   0 -46 -82   3   0   0 -41   2 999
```

Toutes les fonctions sont composées d'une suite de nombres de 2 chiffres. Elles sont identifiées par un nombre négatif supérieur à -80, les nombres négatifs inférieurs à -80 représentant des pointeurs sur des éléments pour déterminer l'(ou les) élément(s) concerné(s). Les autres éléments sont les paramètres et le nombre 999 indique la fin de l'ensemble du VTR.

Les règles de RES1 et de RES2 ont des spécificités différentes par rapport aux règles des TRANS en raison de la particularité de la tâche. La priorité des règles est plus complexe et les conditions peuvent tenir compte de la présence ou de l'absence d'un type d'élément plus à droite dans la phrase.

B) Priorité des règles

Le principe de base est que le *match* le plus long est le meilleur à effectuer car il tient compte de plus de paramètres et d'informations. Mais ce principe est surtout d'actualité pour les *Trans* car la majorité des patterns des règles de RES1 et RES2 ont une longueur égale à 2 (ce qui est également le minimum). Cela implique qu'il est nécessaire de disposer d'un système plus élaboré permettant de déterminer les priorités entre les règles.

Chaque règle dispose de 2 poids et elles sont classées sur base du premier poids, et ensuite, du second. Le premier poids est constitué de la longueur du pattern à laquelle la personne qui écrit la règle peut ajouter 10, 20 ou 30 suivant la priorité qu'il veut lui donner. En général, les règles qui ont une priorité supérieure à 30 effectuent seulement une première passe sur les éléments courants avant de revenir au même élément de départ par l'intermédiaire d'une fonction qui se trouve dans le VTR.

Le deuxième poids est constitué de la complexité sémantique qui est calculée suivant la généralité du pattern et celui qui écrit la règle peut également y ajouter une valeur arbitraire. La complexité sémantique est calculée de la manière suivante :

Pour chaque SWORK du pattern :

- Ajouter 1 si on a une *word class* positive (une *wc*<0 représentant plusieurs *wc*).
- Ajouter 2 si on a un type positif (un type égal à -2 représentant plusieurs types).
- Ajouter 2 si on a au moins une condition.

Exemple

```
32     3  21 43    -1**** -1     0   0  0          64581
```

le chiffre 32 indique que la longueur du pattern est égale à 2 et que la valeur 30 a été ajoutée afin de rendre cette règle fortement prioritaire. La complexité sémantique est égale à 3 pour le premier élément du pattern et 2 pour le second puisque la suite de 4 étoiles fait référence à une condition. En y ajoutant la valeur choisie arbitrairement 1, cela nous donne un total de 6 pour la complexité sémantique (ce chiffre est placé en tête du nombre qui se trouve à la droite du pattern).

Chacun de ces poids est basé sur une valeur calculée par le système et une donnée par celui qui écrit la règle. Cette dualité permet de donner une grande flexibilité à l'attribution des priorités.

C) Recherche vers la droite

Comme l'analyse s'effectue de la gauche vers la droite, on ne sait rien sur les éléments qui se trouvent plus loin dans la phrase. Or, il est parfois nécessaire d'avoir quelques informations sur ce qui se trouve plus à droite afin de prendre la bonne décision à propos de l'élément courant. C'est pour cela que le système Logos a donné la possibilité d'ajouter des conditions, dont le format est similaire aux autres, mais qui sont d'un type particulier. En effet, la satisfaction de ce genre de condition dépend de la présence (ou de l'absence) d'un certain pattern plus à droite dans la phrase.

Exemple

« *The horse run past the barn.* »

« *The horse run past the barn fell.* »

Dans le premier cas, run est le verbe de la principale et dans le second, il est un participe passé. Ce qui nous permet de le savoir, c'est la présence d'un autre verbe plus loin vers la droite.

Le format de ces conditions portant sur des éléments qui se trouvent plus à droite est similaire a celui des autres conditions mais les paramètres font référence à 3 ensembles de règles : *hit rules, skip rules* et *quit rules.* Les *hit rules* détermine les pattern que l'on recherche, les *skip rules* donnent les pattern dont il ne faut pas tenir compte et les *quit rules* représentent les patterns sur lesquels il faut stopper la recherche.

Toutes ces règles se situent dans le même fichier que les règles habituelles. Elles ont donc le même format, c'est-à-dire une suite de SWORK contenant *word class*, *type* et *form* mais ces valeurs ont des significations différentes pour le premier SWORK qui est utilisé comme index. Des *word class* qui n'étaient pas utilisées leur ont été attribuées afin de distinguer les 3 types de règle (07=*hit rule*, 10=*quit rule*, 11=*skip rule*). Le *type* correspond au type de recherche (6300 pour les verbe non ambigü ou 6500 pour les autres). La *form* est un code qui permet de différencier toutes les recherches possibles.

Exemple de règle contenant une recherche vers la droite

```
6300 RULE AT  5 RULE NO. 1512 IS
 22     1  -1 52    -1**** -1     0
  6457 TAGSET STARTING AT ELEMENT    6
  MATCH ON SKIP RULE AT ELEMENT  6 IS   6   11  63 -1   13  -1 -1   -8  -1 -1
  MATCH ON SKIP RULE AT ELEMENT 11 IS   5   11  63 -1   14  -1 -1    1  -1 36
  MTACH ON SKIP RULE AT ELEMENT 15 IS   2   10  63 -1   20  10 -1    0   0  0
```

Cet extrait de diagnostique décrit le déroulement d'une recherche vers la doite. Les éléments du pattern qui représentent l'index sont représentés en gras. La règle qui a fait appel à cette recherche est la suivante (la condition concernée est représentée en gras) :

```
MATCH AT  5 RULE NO. 1512
(V-SIMP PRE-CL)(C7-3)(N U/PV2RT-PL) N(PL) EL(NON-N,WC18)=-1 C7=0 S487
 22     1  -1 52    -1**** -1     0   0  0       0   41710
       6050 1007 8003 1103 8058 1223 8009 9000 6014   82    1   18 9000
       6457   81 8008
       -17   7 -81   1 999
```

Le système va donc prendre un à un tous les éléments qui se trouvent vers la droite jusqu'à ce qu'il trouve une *hit rule* ou une *quit rule* qui *match*. S'il trouve une *skip rule*, il continuera la recherche à partir de l'élément qui se situe après le dernier élément du pattern de la *skip rule*. Si, par contre, il ne trouve aucune règle correspondant au pattern qu'il cherche, il passera à l'élément suivant dans la phrase.

Exemple

« *If XX is open, the YY is closed, and the ZZ is ready, shut the door.* »

Lorsque l'on rencontre la première virgule, on veut savoir si on se trouve dans une série ou si cette virgule constitue la fin de la clause. Pour le savoir, on regarde plus à droite pour tenter de trouver le pattern suivant : « , and ». La règle qui doit être activée pour traiter le cas où on est au début d'une série aura donc la condition suivante :

```
6500   85 8006 8020 888
```

6500 : il s'agit d'une condition portant sur la présence d'un élément vers la droite.

85 : pointeur indiquant qu'il faut commencer la recherche à l'élément suivant le numéro 5.

8006 : indique que l'ensemble des *quit* et *skip rules* est le numéro 6.

8020 : indique que l'ensemble des *hit rules* est le numéro 20.

888 : fin de la condition.

```
6500 TAGSET STARTING AT ELEMENT   11
MATCH ON WC07 RULE AT ELEMENT   11 IS   3    7  20  1   20 888 -1   19  20 -1
```

On constate à la première ligne que le système n'a pas trouvé de pattern correspondant a une *skip, quit* ou *hit rule* avant l'élément numéro 11. La deuxième ligne montre qu'on a trouvé le pattern que l'on cherchait (*hit*=wc07) au onzième élément (ce qui correspond bien à « , and » si on tient compte du *BOS).*

20 888 -1 : représente la virgule.

19 20 -1 : représente le 'and'.

Dans le cas d'une recherche ayant pour condition de succès l'absence d'un certain pattern plus à droite, l'algorithme est le même mais on aura un résultat positif si on trouve une *quit rule* qui *match* avant de trouver une *hit rule*.

D) SEMRES

SEMRES est un ensemble de règles qui sont complémentaires avec celles qui sont utilisées dans les deux parties de RES et qui ont d'ailleurs le même format. Ces règles travaillent principalement sur la morphologie des éléments qui composent le pattern.

L'accès à SEMRES se fait toujours à partir d'une règle du VTR de RES1 ou de RES2 et peut être utilisé soit comme condition au *match* (fonction -22), soit comme un complément à la règle (fonction -18). Dans le premier cas, on ne peut appliquer la règle que si on trouve une règle de SEMRES qui *match* avec les éléments concernés du pattern. Dans le second cas, la règle sera appliquée et, ensuite, on regardera s'il y a un *match* possible dans SEMRES qui impliquera l'application de fonctions supplémentaires.

Afin de faciliter l'accès aux données, l'index utilisé a le même format que les SWORKS. En fait, pour savoir s'il y a une règle de SEMRES qui corresponde au pattern que l'on cherche, on ajoute un élément supplémentaire au pattern qui sert d'index. Ensuite, il ne reste plus qu'à chercher un *match* de la même manière que pour n'importe quelle règle de RES ou des TRANS. Le premier nombre de l'index correspond toujours à la *word class* du premier élément du pattern recherché. Les 2 nombres suivants peuvent être une des 4 combinaisons suivantes (de la plus spécifique à la plus générale) :

- *subset set*
- *set set*
- *set superset*
- *superset superset*

C'est dans cet ordre que le système utilisera ces index afin d'obtenir la règle la plus spécifique possible.

Exemple

« *When the light flashes ...* »

Pour « the light », on a le pattern suivant :

```
14  -1 31     1**** 44
```

Dans le VTR, on a l'appel à SEMRES :

```
-22 -81 -82   0   0   0
```

-22 : *switch* qui fait appel à SEMRES.

-81 : pointeur qui indique que le premier élément du pattern qui doit être trouvé dans SEMRES est le premier élément du pattern de la règle.

-82 : idem pour le second élément.

Les autres paramètres ne sont utilisés que si on a un pattern d'une longueur supérieure à deux. La règle de SEMRES sur laquelle on *match* est :

```
3     14 101  1     14 101 -1     1  -1 42
```

3 : donne la longueur du pattern.

14 101 1 : index basé sur la *word class*, le *subset* et le *set* du premier élément du pattern.

14 101 -1 : premier élément du pattern qui correspond au « the ».

1 -1 42 : second élément du pattern qui correspond au « light ».

Les règles de SEMRES sont utilisées pour des tâches très diverses telles que vérifier l'accord entre un article et un nom, regarder s'il est possible qu'une préposition soit dirigée par un article, si un verbe est à l'impératif, ...

Exemple

« *... these book ...* »

Si cette structure apparaît dans une phrase, c'est par un appel aux règles de SEMRES que le système rejettera la possibilité que 'these' soit l'article de 'book' car il verra que l'un est au pluriel et l'autre au singulier. Cela rendra impossible l'activation d'une règle s'appliquant à un déterminant et au nom avec lequel il s'accorde.

3.5.3. L'application des règles

Le principe qui régit le déroulement des modules *Res* (ainsi que les *Trans*) consiste à analyser une phrase de gauche à droite en effectuant une suite d'itérations qui comportent chacune 2 phases :

- sélection de la règle la plus appropriée.
- application des fonctions qui correspondent à la règle choisie.

Dans un premier temps, il faut donc chercher les règles dont le pattern correspond à la suite d'éléments que l'on est en train d'analyser, c'est ce que l'on appelle le *pattern matching*. Cela signifie que chaque élément du pattern de la règle doit correspondre à une des *part-of-speech* présentes dans le SWORK de la phrase. Dans un second temps, on vérifie que toutes les conditions (les conditions classiques et les conditions concernant des éléments qui se trouvent plus loin dans la phrase) sont respectées et que les accès à SEMRES sont effectués avec succès. Ensuite, on détermine la règle qui possède la priorité la plus importante et on applique les fonctions qui se trouvent dans son VTR.

Exemple de *pattern matching*

Reprenons la phrase suivante :

« *If XX is open, the YY is closed, and the ZZ is ready, shut the door.* »

Le tableau 3.3. indique les informations sous forme de *SAL code* qui ont été obtenues par l'intermédiaire du dictionnaire sur les éléments « *is* » et « *open* ». Le premier dispose de 2 *part-of-speech* possibles correspondant au verbe et à l'auxiliaire tandis que le second en comporte 3 qui sont respectivement le verbe intransitif, le verbe transitif et le nom.

Eléments à analyser	IS	OPEN
1ère *pos*	2 11 3 60 886	2 12 1 31 925
2ième *pos*	12 1 3 60 886	2 7 1 21 925
3ième *pos*		1 13 23 57 180

Tableau 3.3. : Contenu du SWORK pour les éléments « is open »

Le tableau 3.4. montre les 4 règles appartenant à la base de données dont le pattern correspond à une des combinaisons qui peuvent être identifiées dans le tableau précédent et dont les conditions sont respectées. Les pattern des règles 3 et 4 décrivent un auxiliaire suivi d'un verbe. L'échec lors de l'accès à SEMRES indiquera que l'élément « *is* » ne peut pas avoir un rôle d'auxiliaire car il ne s'accorde pas avec « *open* ». Ces 2 règles sont donc automatiquement rejetées.

Règles applicables	Premier élément du pattern	Second élément du pattern	Accès à SEMRES	Niveau de la règle	Complexité sémantique
REGLE 1	2 60 3	2 31 1	NON-REQUIS	22	5
REGLE 2	2 60 3	1 57 23	NON-REQUIS	22	6
REGLE 3	12 60 3	2 31 1	SANS SUCCES	22	7
REGLE 4	12 60 3	2 21 1	SANS SUCCES	22	7

Tableau 3.4. : Règles applicables au pattern « is open »

Il reste donc à déterminer la règle qui est la plus prioritaire entre la règle 1 dont le pattern décrit un verbe suivi d'un autre verbe, et la règle 2 dont le pattern décrit un verbe suivi d'un adjectif. Le premier poids correspondant au niveau de la règle étant le même (22), c'est la valeur de la complexité sémantique de la règle 2 (6) qui les départagera.

La position courante dans l'analyse de la phrase, c'est-à-dire l'élément qui doit se trouver en première place dans le pattern de la règle, est déterminé automatiquement sur base du pattern de la dernière règle qui a été appliquée. En effet, c'est le dernier élément du pattern précédent qui est choisi comme point de départ pour la recherche de la règle suivante. Cette méthode s'assimile à un retour en arrière automatique d'un pas qui est dû à la stratégie choisie par Logos pour l'analyse des phrases au niveau de *Res*. Cette stratégie étant de passer en revue tous les éléments de la phrase en tenant compte de l'élément qui le précède et des informations que l'on a déjà pu acquérir sur ce premier élément. Cependant, il est toujours possible à la personne qui écrit la règle de déterminer un autre élément du pattern comme celui à partir duquel l'analyse doit se poursuivre. Pour cela, il existe une fonction qui peut être ajoutée dans le VTR. S'il n'existe aucune règle qui corresponde au pattern analysé, on continue de la même manière en déplaçant le point de départ de la recherche d'un élément car, dans le cas de *Res*, il n'est pas obligatoire de trouver une règle pour chaque élément.

3.5.4. Les normalisations

Dans le but de réduire le nombre de règles à écrire, certaines informations contenues dans les SWORK (qui proviennent directement du dictionnaire) sont modifiées au début du module RES. Il s'agit principalement de normaliser quelques *word class* telles que les adjectifs et les adverbes. Ces normalisations sont très utiles car il serait fastidieux d'écrire toutes les règles qui correspondent à tous les cas possibles de composition de groupe nominaux ou de groupes verbaux notamment.

Principales normalisations

wc=4 et form=23 (adjectifs)	⇒	wc=1, type=13 et form=23 (nom)
wc=15 (Dét. indéfinis)	⇒	wc=14 (Dét. définis)
wc=2 et form=10,11,12,15	⇒	wc=2 et type=31 (verbe intransitifs)

Tous les éléments retrouveront leurs valeurs initiales à la fin du module RES à l'exception des adjectifs qui garderont une *word class* égale à 1, ce qui correspond aux noms. Cependant, il sera toujours possible de savoir s'il s'agit d'un adjectif grâce à la forme qui reçoit une valeur particulière.

3.5.5. *Rôles linguistiques de RES*

A) RES1

Cette passe ne résout que 2 types d'ambiguïtés qui ont pour particularités d'être sûres et faciles à identifier. En effet, il serait dangereux de faire des erreurs dans cette première passe car les règles appropriées ne pourraient pas être utilisées lors de la phase suivante. Pour sélectionner de manière définitive une *part-of-speech*, les règles disposent d'un *switch* qui peut être inséré dans le VTR.

Le premier cas qui est résolu par la première passe est celui des éléments dont le nom est une des *part-of-speech* possibles et qui sont précédés par un déterminant non-ambigü (un élément est considéré comme non-ambigü s'il ne dispose plus que d'un seul rôle syntaxique possible). Dans ce cas, il est évident qu'il s'agit d'un nom.

Exemple : « ... *the run* ... »
« ... *your house* ... »

Le deuxième cas se présente lorsqu'un élément, qui peut être un auxiliaire, est suivi d'un élément qui peut être un verbe et qu'ils s'accordent (cet accord est vérifié par accès à SEMRES). Ces rôles syntaxiques leurs sont alors définitivement attribués.

Exemple : « ... *should say* ... »
« ... *can be closed* ... »

Le rôle principal de cette première passe est de mettre le maximum d'informations à la disposition de RES2. Il s'agit principalement de faciliter le repérage des verbes de la phrase car

le verbe est l'élément le plus important dans la résolution des homographes. Mais il existe beaucoup d'autres tâches dont RES1 est responsable :

- Lorsque l'on rencontre un verbe qui peut être transitif, RES1 regarde s'il ne trouve pas un objet possible pour ce verbe.
- Marquer ce que le système Logos considère comme des phrases adverbiales

Exemple : « ... *on monday* ... »
« ... *in september* ... »

- Quand on rencontre une virgule, on regarde si on ne se trouve pas dans une série (recherche vers la droite). Cela permet à RES2 de savoir où se termine la clause.
- Modifier le *superset*, le *set*, le *subset* ou la *form* de certains éléments selon les informations recueillies par une règle.

...

Le moyen le plus utilisé pour transmettre ces informations à RES2 est l'attribution des valeurs spécifiques appelées *subchange*. Ses valeurs s'échelonnent de 844 à 866, cette fourchette n'étant pas employée par le subset, cela permet d'utiliser la partie du pattern réservée au type pour indiquer qu'il ne faut appliquer une règle de RES2 que dans certains cas qui ont été identifiés par RES1.

Exemples
WC=2 (Verb) et SUBCHANGE=845 : Verbe non-ambigu.
WC=2 (Verb) et SUBCHANGE=863 : Verbe transitif possible.
WC=2 (Verb) et SUBCHANGE=864 : Verbe transitif faible possible.
WC=19 (Conj.) et SUBCHANGE=860 : 'and' qui se trouve dans une série.

B) RES2

Cette seconde passe a pour rôle de déterminer définitivement les rôles syntaxiques de chaque élément. Pour effectuer pratiquement la sélection du noeud, les règles n'ont plus besoin d'utiliser le *switch* qui permettait à *Res*1 de fixer une *part-of-speech* car cette opération est réalisée automatiquement. C'est la catégorie syntaxique qui correspond à celle qui se trouve dans la règle sélectionnée qui est fixée définitivement. L'analyseur employé par le système est déterministe puisqu'on est certain d'avoir une et une seule *part-of-speech* pour chaque élément à la fin du module RES. La seule exception est le cas où deux noeuds correspondent au pattern de la règle (cela peut arriver avec les *super word class -3 -7 -8 et -9)* mais ces règles ne font

que donner des informations supplémentaires et disposent de retours en arrière qui permettent d'appliquer aux mêmes éléments une autre règle qui déterminera le noeud qui convient.

Il n'est pas obligatoire d'avoir une règle pour chaque élément de la phrase. Si on n'a pas de règle pour fixer une *part-of-speech*, le premier noeud sera choisi par défaut. L'ordre dans lequel les noeuds sont placés dans le dictionnaire est donc très important. La stratégie choisie par Logos est de donner la priorité aux homographes qui sont difficiles a prouver.

Exemple

Dans le cas des articles qui peuvent également être des pronom, il est très difficile de prouver qu'il s'agit d'un pronom. Par défaut, On attribuera donc cette *part-of-speech* à l'élément. De la même manière, on préférera la forme intransitive sur la forme transitive.

Cette passe dispose d'un tableau supplémentaire par rapport à la passe précédente. Il s'agit du tableau CELL qui est composé de 100 entiers et qui permet de connaître l'état courant de l'analyse de la phrase. Les informations que l'on retrouve dans ce tableau sont du type :

- On se trouve dans la proposition principale ou dans une relative.
- On n'a pas encore trouvé le verbe de la proposition dans laquelle on se trouve.
- Le sujet est de type humain, support, ...
- On se trouve dans une phrase impérative, passive, ...
- On a un verbe transitif, on attend donc un objet.

...

L'utilisation du tableau CELL est basée sur la distinction qui est faite entre les groupes considérés comme majeurs, et les groupes considérés comme mineurs. On retrouve les éléments correspondant à ces deux groupes dans différents endroits du tableau CELL suivant leur signification. Cette répartition est illustrée par le tableau 3.5.

Cellules principales		Cellules secondaires		types de groupe
1-10	21-29	51-60	71-79	groupes majeurs
11-19	31-39	61-69	81-89	groupes mineurs

Tableau 3.5. : Répartition du tableau CELL

On distingue deux parties dans le tableau CELL, les cellules principales et les cellules secondaires. Les cellules secondaires qui vont de 51 à 89 sont utilisée pour recopier les cellules principales lorsque la limite d'une proposition est atteinte. Cela permet de garder trace de l'analyse de la proposition précédente.

Les caractéristiques identifiées dans la phrase sont placées dans les cellules correspondant à un groupe majeur lorsque le système identifie les éléments analysés comme appartenant à la proposition principale, à un proposition indépendante ou à une proposition dépendante c'est-à-dire les propositions introduites par une conjonction de coordination. Les groupes considérés comme mineurs sont les propositions relatives, les éléments qui se trouvent à l'intérieur d'une parenthèse, ... Ce classement des types de phrase est très subjectif car les termes utilisés ne correspondent pas toujours à leur définition standard. Logos reprend les termes utilisés dans la grammaire conventionnelle en leur donnant parfois la définition qui correspond à l'utilisation qui en est faite par le système (ce qui s'explique facilement par le fait que ce n'est pas un linguiste qui a écrit les règles de *Res*).

Lorsque la deuxième passe débute, on initialise le tableau CELL afin d'être dans la situation où on se trouve dans la principale (CELL[1]=1) et où l'on attend un verbe (CELL[3]=1). A la fin de ce module, on ne dispose pas d'une représentation claire de la structure de la phrase (arbre, ...) mais une interprétation de l'évolution du tableau CELL permettrait de retrouver la structure identifiée par *Res*.

3.5.6. Exemple de déroulement d'une analyse effectuée par RES

Phrase : *« Write the document ID in the space provided on your Training Information Card. »*

Pour décrire la manière dont cette phrase est analysée, nous allons expliquer, règle par règle, les actions qui sont exécutées. Etant donné que les règles de RES2 tiennent compte d'un grand nombre de conditions (concernant le tableau CELL principalement), seules les plus importantes seront citées. De plus, plusieurs techniques complexes visant à effectuer certaines opérations particulières qui ne sont pas réalisable autrement dans le système actuel ne seront pas détaillée. Le but de cet exemple étant uniquement d'illustrer la manière de travailler du système Logos.

A) RES1

1. *« BOS Write »*
   Pattern: Le *BOS* suivi de n'importe quel élément.
   Cond : On ne se trouve ni dans une phrase interrogative, ni dans une liste.
   VTR : - Attribuer la valeur 900 au subset du *BOS*.
   - Effectuer un double retour en arrière pour reprendre l'analyse au début de la phrase.

2. « *Write the* »

   Pattern : Un verbe transitif suivi d'un groupe nominal.

   Cond : /

   VTR : Attribuer la valeur 862 au subset de *Write* pour indiquer qu'il s'agit d'un verbe transitif possible.

3. « *the document ID* »

   Pattern : Un article défini suivi d'un nom suivi de n'importe quel élément.

   Cond : - Le premier élément est non ambigu (on est certain qu'il s'agit d'un article défini).
   - Le second élément n'a pas dans ses *part-of-speech* possibles un adverbe, un auxiliaire ou un déterminant.

   VTR : - Fixer (définitivement) le second élément comme étant un nom.
   - Effectuer un triple retour en arrière afin de reprendre l'analyse au même endroit.
   - Ne plus utiliser cette règle pour cette partie de la phrase (dû au retour en arrière).

4. « *in the* »

   Pattern : Une préposition suivie de n'importe quel élément.

   Cond : /

   VTR : Attribuer la valeur 2 à la forme du premier élément pour indiquer qu'il peut s'agir d'une préposition ou de la particule d'une verbe

5. « *the space provided* »

   Pattern : Un article défini suivi d'un nom, suivi de n'importe quel élément.

   Cond : - Le premier élément est non ambigu (on est certain qu'il s'agit d'un article défini).
   - Le second élément n'a pas dans ses *part-of-speech* possibles un adverbe, un auxiliaire ou un déterminant défini.

   VTR : - Fixer (définitivement) le second élément comme étant un nom.
   - Effectuer un triple retour en arrière afin de reprendre l'analyse au même endroit.
   - Ne plus utiliser cette règle pour cette partie de la phrase (dû au retour en arrière).

6. « *provided on your* »

   Pattern : Un verbe suivi d'une préposition (CP), suivi d'un groupe nominal.

   Cond : Le second élément n'a pas de nom dans ses *part-of-speech* possibles.

   VTR : Attribuer la valeur 863 au subset du premier élément pour indiquer qu'il s'agit d'un verbe intransitif possible.

7. « *on your* »

   Pattern : Une préposition suivie de n'importe quel élément.

Cond : /

VTR : Attribuer la valeur 2 à la forme du premier élément pour indiquer qu'il peut s'agir d'une préposition ou de la particule d'une verbe

8. « *your Training Information* »

Pattern: Un article défini suivi d'un nom, suivi de n'importe quel élément.

Cond : - Le premier élément est non ambigu (on est certain qu'il s'agit d'un article défini).
- Le second élément n'a pas dans ses *part-of-speech* possibles un adverbe, un auxiliaire ou un déterminant défini.

VTR : - Fixer (définitivement) le second élément comme étant un nom.
- Effectuer un triple retour en arrière afin de reprendre l'analyse au même endroit.
- Ne plus utiliser cette règle pour cette partie de la phrase (dû au retour en arrière).

9. « *Training Information Card* »

Pattern: 3 noms successifs et le premier commence par une majuscule.

Cond : Les 2 noms suivants commencent par une majuscule également.

VTR : /

RES 2

1. « *BOS Write the* »

Pattern: Le *BOS* suivi d'un Verbe transitif possible, suivi d'un groupe nominal

Cond : - La phrase analysée n'est pas de type interrogatif.
- ...

SEM. : Le verbe est sous la forme impérative.

VTR : - *CELL(20)*=20 pour indiquer qu'on se trouve dans une phrase impérative.

2. « *Write the* »

Pattern: Un verbe transitif suivi de n'importe quel élément.

Cond : - Le verbe transitif est suivi d'un déterminant.
- ...

VTR : - Placer le set de *Write* dans *CELL(27)* pour indiquer que l'on a rencontré l'objet du verbe et sauver également dans le même tableau son *superset*, son *subset* et sa *form*.
- Effectuer un double retour en arrière afin de reprendre l'analyse au même endroit.
- Ne plus utiliser cette règle pour cette partie de la phrase (dû au retour en arrière).

3. « *Write the* »

Pattern: Un verbe suivi d'un élément appartenant à un groupe nominal

Cond : - Le second élément n'a pas de pronom relatif dans ses *part-of-speech* possibles.
- ...

VTR : - CELL(7)=3 pour indiquer que l'on se trouve dans un groupe nominal
- Placer dans le tableau CELL les valeurs du superset, du subset et la forme du verbe.

4. *« The document »*

Pattern: un déterminant suivi d'un nom.

Cond : - On ne se trouve pas dans une proposition relative.
- On a trouvé le verbe.
- ...

SEM. : Le déterminant s'accorde avec le nom.

VTR : - Effectuer un double retour en arrière afin de reprendre l'analyse au même endroit.
- Ne plus utiliser cette règle pour cette partie de la phrase (dû au retour en arrière).

5. *« The document ID in »*

Pattern: un déterminant suivi de 2 noms qui sont eux-mêmes suivi d'un élément quelconque.

Cond : - Le quatrième élément ne peut pas être un sujet, un verbe ou un déterminant.
- Le troisième élément ne peut pas être un pronom.
- ...

SEM. : Le déterminant s'accorde avec le second nom.

VTR : - Effectuer un double retour en arrière afin de reprendre l'analyse au même endroit.

6. *« ID in »*

Pattern: un nom suivi de n'importe quel élément.

Cond : - Il n'y a pas de verbe non ambigu plus à droite.
- le second élément ne peut pas être un nom ou un pronom.
- ...

VTR : On supprime la valeur 3 qui se trouvait dans CELL(7) car on vient de trouver la fin du groupe nominal.

7. *« in the »*

Pattern: un déterminant suivi d'un élément appartenant à un groupe nominal.

Cond : - Le premier élément n'a pas d'adverbe dans ses *part-of-speech* possibles.
- ...

VTR : CELL(9) = 13 pour indiquer qu'on a rencontré une préposition

8. *« the space »*

Pattern: un déterminant suivi d'un nom.

Cond : - On ne se trouve pas dans une proposition relative
- On a trouvé le verbe
- ...

SEM. : Le déterminant s'accorde avec le nom.

VTR : - Faire un double retour en arrière afin de reprendre l'analyse au même endroit.
- Ne plus utiliser cette règle pour cette partie de la phrase (dû au retour en arrière).

9. *« space provided on »*

Pattern: 2 noms suivis de n'importe quel élément

Cond : - Le troisième élément doit avoir comme *part-of-speech* possibles un adverbe, une préposition ou une conjonction.
- Le second élément ne peut pas avoir d'adverbe dans ses *part-of-speech* possibles.
- ...

VTR : - CELL(11) = 1 pour indiquer que la suite de l'analyse concerne une proposition et qu'il faut donc utiliser les cellules de tableau CELL correspondant aux groupes mineurs.
- Placer les valeurs du *superset*, du *set*, du *subset* et de la *form* du premier élément dans le tableau CELL.
- Attribuer la valeur 3 à la *word class* du second élément.
- Effectuer un double retour en arrière afin de reprendre l'analyse au même endroit.

10. *« provided on your »*

Pattern: un élément ayant une *word class* égale à 3, suivi d'un déterminant et d'un élément quelconque.

Cond : - On se trouve dans un groupe mineur
- Le troisième élément doit avoir comme *part-of-speech* possibles un nom ou un déterminant.
- Le troisième élément ne peut pas avoir une préposition comme *part-of-speech* possibles.

VTR : Placer les valeurs du *superset*, du *set*, du *subset* et de la *form* du premier élément dans le tableau CELL.

11. *« on your »*

Pattern: un déterminant suivi d'un élément appartenant à un groupe nominal.

Cond : - Le premier élément n'a pas d'adverbe dans ses *part-of-speech* possibles
- ...

VTR : CELL(9) = 13 pour indiquer qu'on a rencontré une préposition

12. *« your Training »*

Pattern: un déterminant suivi d'un élément appartenant à un groupe nominal.

Cond : On a trouvé le verbe.

SEM. : L'accord entre les 2 éléments du pattern est correct.

VTR : - Attribuer la valeur 10 à la *form* du premier élément pour indiquer qu'il peut être singulier ou pluriel.
- Effectuer un double retour en arrière afin de reprendre l'analyse au même endroit.
- Ne plus utiliser cette règle pour cette partie de la phrase (dû au retour en arrière).

13. *« your Training Information »*

Pattern: un déterminant suivi de 2 noms.

Cond : - Le deuxième élément n'a pas de déterminant dans ses *part-of-speech* possibles.
- Le deuxième et le troisième élément débutent par une majuscule.

VTR :/

14. *« Information Card EOS »*

Pattern: 2 noms suivis de n'importe quel élément.

Cond : Le premier et le deuxième élément débutent par une majuscule.

VTR : Effectuer un double retour en arrière.

15. *« Card EOS »*

Pattern: Un nom suivi de n'importe quel élément.

Cond :/

VTR : - Effectuer un double retour en arrière afin de reprendre l'analyse au même endroit.
- Ne plus utiliser cette règle pour cette partie de la phrase (dû au retour en arrière).

3.6. Les modules Trans

3.6.1. But

Le but des modules *Trans* est de générer, phrase par phrase, la traduction du texte source dans la (ou les) langue(s) cible(s) choisie(s). Pour cela, ils disposent au départ de la traduction par défaut de chaque élément de la phrase (traduction obtenue par l'intermédiaire du dictionnaire lors de l'application du module Gerdem) plus leurs rôles syntaxiques qui ont été déterminés par *Res*. L'analyse s'effectue en 4 passes qui disposent chacune d'un ensemble propre de règles et qui peuvent toutes faire référence à un ensemble commun (SEMTAB). Les fonctionnalités des *Trans* ne se limitent cependant pas à agir sur la phrase cible. En effet, les 3 premières passes effectueront des simplifications dans la phrase source afin de faciliter l'écriture des pattern des règles utilisées par les phases suivantes.

Qu'elles s'appliquent aux phrases sources ou aux phrases cibles, les règles appartenant aux modules *Trans* utilisent chacune un grand nombre de fonctions (ceci est dû au fait que ces fonctions sont toutes de très bas niveau) et sont souvent communes à plusieurs règles. Ces fonctions ont donc été regroupées dans des tables qui peuvent (comme les règles) être soit propres à un module, soit être communes à l'ensemble des *Trans*. Il existe 3 types de tables :

- *30 tables* : agissent sur la source uniquement et sont propres à un module.
- *40 tables* : agissent sur la source ainsi que sur la cible et sont propres à un module.
- *50 tables* : agissent sur la cible uniquement et sont communes aux 4 modules.

Exemple de règle comprenant un appel à une 30 table

```
1***** A MATCH STRATING AT    6 LEVEL 2                          ON ELEMENT  6JJ
N(91) NON-N = -2 / CK S3                                        STS586 EGSP1
      2      1  -1  91        -3  -1  -1         0   0   0
                -20    0 -63    1 352    3 -36  56    0 -41    2 999

MAIN 30 TABLE
     -55    3 -81 351 -56    1 399  56    3  18 888 888
     -66 123  56 -81    3    9  60 -66 299 399 -81    3    6  60 888 888
     -57    1 -54    1 -81    3    1 -48  13    3 -81 -46 -81    0   0    2 888 888
     -57    2 -54    1 -81    3    1 -48  13    3 -81 -46 -81    0   0    1 -57    3 999
CELL  3  =     1
-56 056 CONDITION AT    9,                                  CONTINUE TO THE RIGHT
SCON(  3,-81) =     3
-66 056 CONDITION AT   14,                                  CONTINUE TO THE RIGHT
SCON(  3,-81) =     3
-66 SWITCH TEST: CONDITION FALSE
BRANCH TO -57  3    EXECUTE UNTIL -57 99  JUMP -57 99
```

L'appel à la *30 table* est effectué par le Switch -63 qui est en gras ci-dessus.

3.6.2. Les règles

Les caractéristiques des règles appliquées lors des modules *Trans* sont un peu différentes des règles de *Res* mais elles disposent d'un format identique (commentaires, pattern, conditions, fonctions à appliquer). Elles ne peuvent pas effectuer de recherches vers la droite mais disposent d'un système de stretchs et elles travaillent sur des tableaux différents. De plus, l'ensemble commun de règles s'appelle SEMTAB et son rôle est bien sur différent de SEMRES même si le principe est similaire.

A) SEMTAB

SEMTAB est l'équivalent de SEMRES pour les modules *Trans* mais il traite des informations différentes. Cela signifie qu'il peut être accédé à partir des 4 modules *Trans* et que le format des règles est le même. On ajoute donc également un élément au début du pattern de chaque règle afin de s'en servir comme index et cet élément est construit à partir de l'élément considéré comme clé du pattern.

Exemple de règle contenant un appel à SEMTAB

```
3***** A MATCH STARTING AT   3 LEVEL   3                    ON ELEMENT  4JJ
*28 V .S. PUNC = -A*O(SIC ) / SMTB                        STS586 EGSP3
      3     2  -1  -1       52  -1  -1       20  -1  -1
                  -20   0 -22   4 -81   1  -1   0 -99  91  -2   0  -3   0 -46 -81
                   17   0   0 -41 100 999
STR1CHG:   1          STR2CHG:   0          TSR3CHG:   0
VTRF LOADED  -20   0 -22   5 -81   1  -1   0 -99  91  -2   0  -3   0  -4   0
             -46 -81  17   0   0 999
VTRF LOADED  -20   0 -22   5 -81   1  -1   0 -99  91  -2   0  -3   0  -4   0
             -46 -81  17   0   0 999
     ***SEMWRK VALUES
        2  886   60    0         2  886    3    4         1   27   91    3
       13  522    3    5         1   23    1    6        20   10    1    7
        SEMTAB MATCHING PARAMETERS HAVE BEEN LOADED AS FOLLOWS:
        LOGUSR =   1 USRUSR =   2EXTENDED SEARCH =   1 LUDIFF =   1
        EL1LVL =   2 CMPEL1 =   2 CMPLEX =   2
        CCLIST(1-5) = CMG
        *&  NO MATCH FOUND

SEMWRKS =    1  2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
NSWORKS =    1  3  2  4  5  6  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 DIAG      3    6    3    1    0    0    0
        SWORKO   3      7  27  17  3
```

SEMTAB a plusieurs fonctions mais la plus importante est de déterminer la traduction qui est la plus appropriée pour un élément suivant le contexte sémantico-syntaxique dans lequel il se trouve. De la même manière qu'il s'agit de la *part-of-speech* la plus difficile à prouver qui est choisie par défaut, c'est la traduction dont la nuance est la plus difficile à déterminer qui sera placée dans le dictionnaire (ce choix peut être révisé dans les cas où une stratégie différente s'est avérée plus payante dans la pratique). Comme en plus de cela il existe dans ce module des règles qui permettent de corriger les erreurs réalisées par *Res* à cause du nombre limité de *part-of-speech* présentes dans le dictionnaire, SEMTAB est considéré comme une extension de ce dictionnaire.

Parmi les autres fonctions de SEMTAB, on peut mettre en évidence la détection de certaines caractéristiques syntaxiques (forme des verbes, type des noms, lien entre objet et verbe, ...) qui seront utilisées dans la suite de l'analyse.

B) Les stretchs

Un *Stretch* est un élément de type particulier qui peut être inséré dans le pattern d'une règle. Il représente n'importe quelle liste d'éléments (cette liste pouvant être vide) qui se trouvent dans la phrase à analyser. En effet, il arrive souvent qu'une bonne traduction nécessite de tenir compte de 2 mots qui ne se trouvent pas impérativement l'un à la suite de l'autre et qu'il soit impossible de déterminer le nombre ainsi que le type des éléments qui peuvent se trouver au milieu. Cependant, il est possible de définir des conditions quant aux éléments qui peuvent se trouver dans le stretch.

Exemple de pattern contenant un stretch

```
**175 THAT(REL) N(OBJ OF PRP) N(SUBJ) .S. V .S. PRP EL(^N)=-A*2(NUL PRP) S29
      S    20 103  18        1****  94        1  -1  91      1041 1806
           51****  -1        2  -1  -1       51****  -1       13  -1  -1       -1****  -1
   TAG: 6013 0081 0020 0000 0000 0000 0000 0000 0000 0000
           0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
   TAG: 6014 0088 0001 7777 6088 2072 8012 2072 8013 2072
           8014 0000 0000 0000 0000 0000 0000 0000 0000
```

Les stretchs sont identifiés par la présence d'une *word class* égale à un nombre compris entre 50 et 59. Chacun de ces stretchs ayant des caractéristiques particulières au niveau des conditions. Les conditions relatives aux stretchs se trouvent à la suite des mots *TAG*.

Ce mécanisme est utilisé dans des situations très diverses. Par exemple, lorsqu'une règle veut traiter le mot « some » et le nom auxquel il se rapporte, elle doit tenir compte du fait que d'autres éléments peuvent s'intercaler comme dans « ... some very important decisions ... ».

C) Les tableaux

Les linguistes qui écrivent les règles peuvent, grâce aux fonctions qui se trouvent dans le VTR, stocker leurs informations dans 3 tableaux (SCON, CELL, SLOT) qui ont des portées et des rôles différents.

Le tableau le plus important est appelé SCON (*Semanto-syntactic CONtrol array*). Il contient la grande majorité des informations qui sont recueillies lors de l'application des différentes règles des 4 *Trans*. Il est formé de 400 lignes et 110 colonnes d'entiers. Chacune des 100 premières lignes du tableau est réservée à un élément de la phrase (la x<u>ième</u> ligne correspondant au x<u>ième</u> élément) tandis que chaque colonne correspond à un type d'information pré-déterminé (il existe, par exemple, une colonne pour les *word class*, une pour

le nombre, ...). Dans les lignes allant de 100 à 120, on trouve les VC's (*Variable Constant*) qui sont utilisés pour stocker des informations relatives aux éléments qui n'existaient pas dans la langue d'origine mais qui sont nécessaires dans la langue cible ou qui doivent être déplacés. Toutes les autres lignes sont réservées pour des constantes ayant des rôles très divers.

Le tableau CELL, déjà utilisé lors de la seconde passe du module *Res*, est formé de 100 entiers. Il peut être utilisé comme espace de travail temporaire ou comme lieu de stockage pour des informations telles que le format ou la forme d'une phrase. Ce tableau est divisé en trois parties qui ont chacune une portée différente (1-10: une règle, 11-39: un *Tran*, 40-99: tous les *Trans*).

Le dernier tableau se nomme SLOT et est utilisé pour placer une suite de commandes qui ne peuvent pas être exécutées immédiatement. C'est plus tard dans l'analyse (il peut même s'agir d'un autre *Tran*) que l'on effectuera ces fonctions.

3.6.3. L'application des règles

Le principe utilisé lors de l'application des *Trans* est similaire à celui qui est employé par le module *Res* mais il existe cependant certaines différences notamment au niveau du retour en arrière. La stratégie utilisée par le système Logos lors de *Res* était d'analyser la phrase en étudiant principalement les éléments deux par deux. Cette méthode obligeait à réaliser un retour en arrière chaque fois qu'une règle avait été appliquée. Dans le cas des *Trans*, le but est de trouver la traduction correcte de chaque élément à l'intérieur de son groupe syntaxique. Il n'est donc pas nécessaire d'effectuer de retour en arrière. De plus, il est obligatoire de trouver une règle à appliquer pour chaque élément, il existe donc certaines règles qui sont utilisées par défaut.

L'ordonnancement des règles et l'algorithme utilisé pour y accéder permettent de gérer automatiquement les priorités. La stratégie des priorités pour les *Trans* est la suivante : on prend d'abord la règle la plus spécifique et, ensuite, celle qui a le pattern le plus long. Le critère de spécificité est lié aux types des éléments qui composent le pattern de la règle. En effet, un pattern comprenant un *subset* est plus spécifique qu'un pattern dans lequel on trouve un *set* ou un *superset*.

Un système de double index placés respectivement sur les 2 premières *word class* permettent de trouver rapidement les ensembles de règles susceptibles d'être applicables au pattern courant. De plus, à l'intérieur de ces ensembles, ces règles sont placées d'abord par ordre de

spécificité et, ensuite, suivant leur longueur. Il suffit donc de les parcourir dans l'ordre et la première qui *match* entièrement sera également la plus prioritaire.

Exemple

Supposons que l'on cherche une règle s'appliquant aux éléments : « *article on treaty* ». Les valeurs du *SAL code* qui se trouvent dans le tableau SWORK sont détaillées dans le tableau 3.6.

SAL code des éléments à analyser		
article	on	treaty
1 375 1 76 12	13 214 3 44 3	1 522 1 38 07

Tableau 3.6. : Contenu du SWORK pour les éléments «article on treaty »

Le tableau 3.7. décrit l'ensemble des règles auxquelles le système peut accéder. Seules les règles qui sont concernées par la recherche qui nous occupe sont présentes. Le premier accès à l'ensemble des règles se fait au niveau des règles 9 et 10 car elles concernent les caractéristiques les plus spécifiques du mot « *article* », c'est-à-dire le *subset* qui est égal à 375. Les index placé sur les 2 premières *word class* des règles indique immédiatement qu'il n'existe aucune règles pour cette valeur du *subset*. Le deuxième accès est effectué (au niveau des règles 6, 7 et 8) sur base du *set* qui vaut 76. Cet accès échouera également puisque la seule règle dont le premier élément du pattern correspond a celui recherché diffère au niveau du deuxième élément. Le troisième accès réalisé sur base du *superset* sera le bon car il existe 2 règles (3 et 4) dont le pattern correspond aux éléments recherchés. Le système analyse les règles de bas en haut afin de respecter l'ordre de spécificité et de longueur. La première analyse concerne donc la règle 3 mais le *type* du troisième élément de celle-ci ne correspond pas. C'est donc la règle 3 qui sera appliquée.

Ensemble des règles			
règles	Premier élément du pattern	Deuxième élément du pattern	Troisième élément du pattern
1	1 -1 -1 ...		
2	1 7 1	13 103 2	1 1 -1
3	1 12 -1	13 214 3	
4	1 12 -1	13 214 3	1 722 -1
5	1 13 -1 ...	...	
6	1 75 -1	...	
7	1 76 -1	14 -1 -1	
8	1 77 -1 ...	...	
9	1 370 -1	...	
10	1 380 -1 ...	...	

Tableau 3.7. : Règles applicables au pattern « article on treaty »

La règle numéro 1 est appelée règle par défaut. Elle est appliquée dans le cas où aucune règle ne correspond au pattern recherché. Il existe une règle par défaut pour chaque *word class* possible. De cette manière, il est certain qu'une règle correspondra qu'elle que soit le pattern présent dans la phrase analysée.

Grâce à cette méthode, une forte augmentation du nombre de règles n'entraîne pas une diminution proportionnelle des performances car ces 2 caractéristiques ne sont pas liées par une relation linéaire. La stratégie de Logos au sujet de la gestion des règles à donc été de créer un très grand nombre de règles simples et spécifiques.

3.6.4. Rôles linguistiques des quatre Trans

Chaque *Tran* doit effectuer plusieurs tâches au niveau linguistique sur la phrase source et sur la phrase cible. En ce qui concerne les actions sur la phrase source, chaque module recherche un certain type de groupe syntaxique afin de le concaténer, cela permet de faciliter l'écriture du pattern des règles utilisées par les modules suivants.

Cependant, la répartition des rôles n'est pas très claire. La méthode utilisée pour définir ces rôles dépendant uniquement de la nécessite d'effectuer des opérations préliminaires, aucune structure n'a été établie pour attribuer à chaque module l'ensemble des opérations d'un même type.

A) Tran1

*Tran*1 a pour but d'identifier le temps, la voix, le mode et le sujet du verbe principal. Il s'occupe surtout des groupes nominaux simples mais il doit aussi détecter et/ou concaténer, les groupes verbaux de la principale et les groupes verbaux qui contiennent un auxiliaire. Dans ce dernier cas, la *form* du verbe reçoit une valeur dont on peut déduire la nature du verbe (actif modal, passif modal, passif simple, ...).

Il accède à SEMTAB pour effectuer 2 tâches. La première est de corriger les erreurs de *Res* qui sont dues à la limitation à 3 du nombre de *part-of-speech* existant dans le dictionnaire. La seconde consiste à effectuer des modifications sémantiques pour adapter la traduction au contexte.

B) Tran2

Les tâches linguistiques de *Tran*2 concernent principalement les propositions. Celles-ci sont placées à la fin de la phrase pour être analysées séparément. *Tran*2 détecte et/ou concatène les phrases prépositionnelles, les propositions relatives, les noms composés et les phrases qui contiennent un participe passé.

Il utilise également SEMTAB pour modifier la traduction de certaines propositions relatives et les groupes nominaux composés qui sont ensuite concaténés. Par conséquent, tous les groupes nominaux ne seront plus représentés que par leur nom principal à la fin de ce module.

*Tran*2 a la possibilité de corriger certaines décisions qui ont été prises lors du module précédent mais il s'agit d'une option qui peut être supprimée par la personne qui écrit les règles.

C) Tran3

Il complète le travail de *Tran*2 sur les propositions, principalement en ce qui concerne les propositions imbriquées. Il est spécialisé dans les travaux ayant rapport avec la sémantique des verbes et la gestion des prépositions. Ses différents rôles sont de :

- concaténer les compléments du verbe.
- effectuer des transformations sémantiques sur les verbes (sur base de leurs objets, de leur(s) complément(s) ou de leur sujet).
- effectuer des transformations sémantiques sur les compléments des verbes.

D) Tran4

*Tran*4 prend les décisions finales en se basant sur la phrase dans son ensemble. Son travail est plus hétérogène, les tâches qui lui soit attribuées sont plus spécifiques que dans le cas des modules précédents. Parmi ces tâches, on peut souligner :

- utiliser le subjonctif.
- passer de la voix passive à la voix active.
- réordonner les éléments qui forment une proposition.
- effectuer les transformations sémantiques sur les verbes qui n'ont pas pu être réalisées par le module précédent.
- corriger éventuellement certaines erreurs commises auparavant.

3.7. Le module Fprint

Le module Fprint a pour rôle de transformer la représentation interne de la phrase reçue de *Tran*4 et d'insérer des séquences d'échappement faisant référence au format de sorte que Postproc puisse établir le document final en effectuant l'opération inverse de celle qui a été réalisée par Preproc. Cependant, Fprint doit également effectuer quelques transformations linguistiques telles qu'appliquer les élisions, les délitions, ...

Fprint reçoit la traduction sous la forme d'une suite d'adresses qui font référence à un groupe de dictionnaires. L'identification du dictionnaire contenant l'adresse concernée est aisé puisqu'ils disposent chacun d'une fourchette différente d'adresses :

1 → 70 : dictionnaire principal
> 131 : *low frequency constant dictionnary*
< -131 : *high frequency constant dictionnary*

Les nombres restant correspondent aussi à des éléments particuliers mais ils ne se trouvent pas dans un dictionnaire :

-11 → -37 : mots inconnus
-121 → -129 : ponctuation

Une fois que l'accès au dictionnaire adéquat est effectué, 2 cas peuvent se présenter : l'adresse correspond soit à un mot ne devant pas être fléchi, soit à une racine. S'il s'agit d'un mot, il suffit

de placer la suite de caractères dans le fichier de sortie. Si, par contre, il s'agit d'une racine, il faut accéder aux *PAT tables* (l'équivalent pour le langage cible de celles qui sont utilisées par Gerdem). Le dictionnaire dispose, pour chaque entrée, du numéro de la table qui correspond à la racine. Les informations morphologiques qui se trouvent dans le tableau SCON sont utilisées comme clés d'accès pour les *PAT tables*. Le système en déduit la manière de fléchir la racine et insère dans le fichier de sortie la suite de caractères qu'il a générée.

La phrase générée doit encore subir quelques opérations grammaticales pour être correcte car les modules *Trans* ne prennent pas en charge la réalisation des élisions, des contractions, des délitons, et des condensations.

exemples

élision : je → j'
contraction : a les → aux
délition : les plus meilleures → les meillleures
condensations : m'aime il → m'aime-t-il

NB : Les termes utilisés ne correspondent pas exactement à leurs définitions linguistiques rigoureuses.

Lorsque ces 4 opérations ont été effectuées, le processus de traduction en lui-même est terminé puisque la phrase se trouve sous sa forme définitive. Il reste cependant à les placer dans un fichier *LTX* afin que Postproc puisse recréer le document final selon un format identique à celui du document initial. Cette tâche comprend, en plus de l'écriture des caractères représentant la phrase traduite, l'insertion des séquences d'échappement permettant de faire référence au tableau contenant le format. Cette tâche n'est pas aussi triviale qu'il y paraît car l'ordre des mots peut varier au cours du processus de traduction (ce problème est résolu par la présence de pointeurs dans le tableau SCON qui mémorisent la position initiale de l'élément et qui permettent de faire référence au tableau contenant le format créé par Gerdem) et de nouveaux mots peuvent être insérés.

3.8. Le module Postproc

La seule tâche de ce module est de créer le document final selon un format identique à celui du document initial. Pour cela, il se base sur le fichier créé par Fprint contenant la traduction ainsi que sur le fichier *Logos Markup File*, généré par Preproc, qui décrit le format.

4. LES OUTILS FOURNIS AUX UTILISATEURS

Afin de ne pas être limité à certains domaines particuliers, l'élaboration des règles se fait sous l'hypothèse que le système doit être *general purpose*. Cependant, les clients ont des besoins spécifiques et il est nécessaire qu'il puissent adapter le programme Logos à leurs domaines de travail. L'utilisation des *subject matter code* étant insuffisante pour satisfaire ce besoin, Alex (point 4.1) et Semantha (point 4.2) ont été mis à la disposition des clients.

Ces 2 logiciels ne requièrent aucune connaissance des *SAL codes* et aucune expérience de programmation. Les seules compétences que doit avoir l'utilisateur de Alex est de Semantha se situent au niveau linguistique. Malgré tout, les clients ne peuvent pas réaliser toutes les modifications eux-mêmes. Certaines sont trop compliquées ou nécessitent une connaissance plus approfondie de la manière de travailler du système. Dans ce cas là, les utilisateurs peuvent faire appel à la section *customer support* de Logos (il arrive d'ailleurs que Alex ou Semantha indique directement que l'opération que veut réaliser le client doit être soumise à cette section).

Le but d'Alex n'est pas de permettre à l'utilisateur d'élaborer un dictionnaire entier mais de lui permettre d'adapter le dictionnaire existant. C'est pour cela qu'il est limité dans les actions qu'il peut effectuer. Par exemple, il ne peut pas modifier les informations concernant des articles, il peut ajouter une *part-of-speech* pour un verbe existant mais on ne peut pas en créer un nouveau, ...

Nous allons maintenant expliquer la démarche qui doit être suivie par les utilisateurs qui désirent modifier soit le dictionnaire, soit SEMTAB. Pour cela, nous ne tiendrons compte que de la paire de langues suivante : anglais-français. En effet, la démarche est très similaire pour les autres cas mais il existe évidemment certaines spécificités pour chaque paire de langues possible.

Les informations présentées ci-dessous sont extraites de [MANUAL] qui est le manuel de référence fourni avec le système dont la langue source est l'anglais.

4.1. Alex

Alex est un programme qui permet à l'utilisateur de modifier le dictionnaire principal (le dictionnaire *high frequency* ne peut pas être modifié). Pour cela, le produit offre 3

fonctionnalités qui permettent de gérer les entrées : ajouter, supprimer et visionner une entrée. De plus, il est possible de demander 2 types de rapport.

4.1.1. Ajout d'une entrée

La démarche à suivre pour ajouter une entrée au dictionnaire est différente suivant la catégorie syntaxique de l'élément considéré. Dans un premier temps, il faut donc choisir parmi les possibilités suivantes : nom, adjectif, adverbe ou verbe. La *part-of-speech* sera automatiquement déduite de ce choix.

Cependant, certaines informations sont communes à tous les cas. Il s'agit du mot source, de sa traduction, du *subject matter code* et du *company code*. La démarche particulière à suivre pour les différentes *part-of-speech* vise surtout à déterminer le *SAL code* qui doit être attaché à la nouvelle entrée car, comme nous l'avons dit ci-dessus, il n'est pas utile que l'utilisateur maîtrise les codes internes du système.

A) Nom

Les informations spécifiques qu'il faut entrer pour insérer un nom sont le genre, le nombre, l'adjectif correspondant au nom et, s'il s'agit d'un élément comprenant plusieurs mots, le *head noun* (c'est-à-dire celui qui est considéré comme l'élément principal).

Pour qu'Alex puisse déterminer le *SAL code*, l'utilisateur doit entrer un nom qui appartienne à la même catégorie. Le système attribuera donc le *SAL code* de ce mot au nouveau nom. Dans le cas particulier d'un élément comprenant plusieurs mots et qu'il existe déjà une entrée pour le *head noun*, le nouvel élément recevra automatiquement le *SAL code* de cette entrée.

La référence aux *PAT-tables* (contenant les inflexions) est déterminée par un programme appelé *stemgem* qui se base sur les réponses données par l'utilisateur au sujet de la morphologie de la nouvelle entrée.

B) Adjectif

Dans ce cas-ci, les informations spécifiques sont la traduction de l'adverbe correspondant et le fait que l'adjectif soit de type *pre-posed* ou *post-posed*. Quant à la définition du *SAL code*, l'utilisateur doit choisir parmi 5 possibilités :

- forme passée d'un verbe.
- adjectif pre-clausal
- adjectif pre-verbal
- préfixe
- autre

Ensuite, l'utilisateur doit déterminer parmi une liste d'exemple lequel est le plus proche du comportement de l'élément qu'il veut ajouter. Pour faire son choix, il peut s'appuyer sur la manuel dans lequel se trouve une description plus approfondie de chaque possibilité ainsi qu'une liste plus détaillée d'exemples.

C) Adverbe

La démarche est très semblable à celle qui doit être suivie pour les adjectifs car il faut également donner la traduction de l'adjectif correspondant et dire s'il s'agit d'un adverbe *pre-posed* ou *post-posed*. Cependant, il n'y a pas de choix préliminaire à faire pour la définition du *SAL code* car il faut directement déterminer parmi 9 exemples lequel correspond le mieux à la catégorie du nouvel élément.

D) Verbe

En ce qui concerne les verbes, l'utilité de Semantha se situe au niveau de la modification de la traduction par défaut. Afin de distinguer l'entrée (qui doit déjà se trouver dans le dictionnaire) à laquelle la modification doit s'appliquer, le logiciel demande le verbe ainsi que quelques informations concernant la réflexivité, la transitivité et l'auxiliaire qu'il requiert.

4.1.2. Supprimer une entrée

Alex permet évidemment de supprimer une entrée mais aussi de la désactiver, la réactiver ou l'exporter. La désactivation permet de voir l'effet de l'absence d'une entrée sur une traduction sans être obligé de la supprimer physiquement. L'exportation consiste à copier une entrée d'un code (*subject matter code* ou *company code*) vers un autre.

4.1.3. Les rapports

Il existe 2 types de rapport. Le premier permet d'accéder aux entrées selon certains critères définis par l'utilisateur. Le second est une sorte de journal regroupant toutes les opérations qui ont été effectuées depuis la dernière sauvegarde des modifications réalisées.

4.2. Semantha

Semantha est un logiciel qui permet à l'utilisateur de gérer les règles de SEMTAB utilisées par les modules *Trans*. Grâce à ce programme, il est possible de modifier la traduction d'un élément selon des critères syntaxiques et sémantiques.

La question qui se pose naturellement à l'utilisateur lorsqu'il désire modifier la traduction d'un élément est de savoir quel outil est le plus approprié dans ce cas (Alex ou Semantha). Pour faire son choix, il doit se baser sur deux critères : la provenance de la traduction actuelle et la particularité du contexte syntaxico-sémantique.

Exemple

Supposons que la traduction du verbe « to file » qui se trouve dans le dictionnaire soit « ranger ». Si le client désire modifier la traduction dans tous les cas, il utilisera Alex. Si, par contre, il veut uniquement modifier la traduction dans une phrase telle que « to file a patent », il se servira de Semantha pour écrire une règle qui conduira à la traduction par « déposer » de toutes les occurrences verbe « to file » s'appliquant au nom « patent ».

Semantha donne la possibilité d'ajouter, de visionner et de supprimer des règles mais il met aussi 3 outils à la disposition de l'utilisateur.

4.2.1. Ajouter un règle

Après avoir choisi la langue cible pour laquelle la règle devra s'appliquer, l'utilisateur doit choisir parmi 8 *templates* exprimés grâce à un principe similaire à celui qui est employé pour les commentaires que l'on trouve dans chaque règle. Pour les traductions de l'anglais vers le français, le choix est le suivant :

- Verb Noun/Adjective = Verb (Prep) Noun/Adjective
- Verb Prep Noun = Verb (Prep) Noun

- Verb Noun1/Adj Prep2 Noun2 = Verb (prep1) Noun1/Adj (Prep2) Noun2
- Verb-Particle Noun = Verb (Prep) Noun
- Verb-Particle Prep Noun = Verb (Prep) Noun
- Verb-particle Noun1 Prep2 Noun2 = Verb (Prep1) Noun1 (Prep2) Noun2
- Noun1 Prep Noun2 = Noun1 Prep Noun2
- Descriptive Adjective Noun = Adjective Noun

Tout comme pour Alex, des exemples sont donnés à l'écran et des informations complémentaires se trouvent dans le mode d'emploi. Ensuite, il ne reste plus qu'a donner quelques informations de base sur les différents composants du pattern sélectionné.

4.2.2. Visionner une règle

Cette fonctionnalité affiche à l'écran toutes les caractéristiques d'un ensemble de règles. Pour déterminer les règles qu'il veut visionner, l'utilisateur doit donner une *part-of-speech* d'un élément. Le programme recherchera automatiquement toutes les règles qui sont en relation avec cette entrée.

4.2.3. Supprimer une règle

Comme dans le cas d'Alex, cette fonctionnalité ne comporte pas seulement le retrait physique de la règle mais aussi la possibilité de désactiver et de réactiver une règle.

4.2.4. Les outils

Le premier outil consiste en une sauvegarde automatique dans un fichier de tous les changements qui ont été effectués. Le second permet d'accéder à toutes les informations se trouvant dans ce fichier.

Enfin, il existe une fonctionnalité permettant d'écrire dans un fichier toutes les règles qui appartiennent à un certain *company code*.

5. AMELIORATIONS FUTURES

Nous allons présenter 3 projets développés par la compagnie Logos afin d'améliorer la qualité du système. Ils sont tous les 3 à différents stades de leur réalisation. Le premier concernant la *translation memory* (point 5.1) est sur le point d'aboutir, le second traitant du *translatability Index* (point 5.2) est en plein développement (il s'agit d'ailleurs du projet sur auquel nous avons participé), et le troisième ayant rapport avec les *Semcodes* (point 5.3) n'est encore qu'au stade théorique.

5.1. Translation Memory

Actuellement, Logos Corporation met au point en collaboration avec Eurolang un environnement de travail pour la traduction intégrant Optimizer d'Eurolang et le système de traduction Logos. Afin de répondre aux besoins des utilisateurs, il a fallu ajouter au système Logos certaines fonctionnalités de Translation Memory. Ci-dessous, nous allons expliquer le principe de la Translation Memory et voir comment de telles techniques peuvent contribuer à l'élaboration d'un environnement de travail augmentant considérablement semble-t-il la productivité du traducteur.

Le lecteur désirant avoir de plus amples informations sur ce sujet peut se référer à [SCOTT_d].

5.1.1. Principe

Les outils de Translation Memory qui apparaissent sur le marché depuis 6 ou 7 ans sont l'expression, en Traduction Automatique du simple principe d'économie « Ne pas refaire ce qui a déjà été fait ». En l'occurrence, il s'agit de ne pas retraduire ce qui a déjà été traduit. Cette façon de faire se justifie d'autant plus lorsqu'on pense aux cas relativement fréquents de documents déjà traduits dont il faut traduire une nouvelle version dans laquelle nombre de phrases sont restées inchangées par rapport à l'original.

Dans le cas de Logos, le principe de Translation Memory est appliqué à deux niveaux :

Premièrement, au début de la traduction, les phrases sont comparées à celles qui se trouvent dans une base de données spécifiée par l'utilisateur en fonction du document. Il s'agit ici de la forme la plus simple de Translation Memory. Tout ce qu'il y a à faire c'est d'enlever de

l'ensemble des phrases à passer au système celles qui se trouvent dans la base de données et de fournir à l'utilisateur la traduction qui en a déjà été faite précédemment. On a raffiné quelque peu cette approche en faisant la distinction entre les phrases qui sont restées inchangées (*perfect match*, c'est-à-dire 100% de similitude) et celles qui ont été légèrement modifiées (*fuzzy match*, c'est-à-dire environ 90% de similitude). Les *fuzzy matches* sont quand même passés à la traduction et ce sera à l'utilisateur de voir ce qu'il en garde.

Logos essaie également d'établir des correspondances avec des structures de niveau inférieur à celui de la phrase. Actuellement, seules les propositions nominales sont traitées. Dans l'environnement de post-editing, lorsque l'utilisateur édite la traduction d'une proposition nominale (la plupart du temps parce que le système n'a pas bien identifié les interactions sémantiques des mots qui la composent ou parce que la véritable traduction est une « expression consacrée » de la langue cible ou fait partie de la terminologie de l'utilisateur), celle-ci est sauvée dans la NP-memory. Si, par la suite, un autre document est traduit, à la fin de la traduction, les propositions nominales sont recherchées dans la NP-memory. Cette manière d'utiliser la Translation Memory peut être vue comme un forme primaire[5] de Machine Learning, car c'est au fur et à mesure des utilisations que la machine « apprend » les fautes à ne plus commettre. Le problème principal de la NP-memory est que si la langue cible est fortement fléchie, il faut pouvoir générer les diverses inflexions des mots contenus dans les propositions. Pour cette raison, le choix a été fait de stocker les entrées de la NP-memory sous forme canonique, de leur faire correspondre une traduction également sous forme canonique. Ceci implique que tant pour établir la correspondance avec une entrée que pour générer son transfert, il faudra faire usage d'information morphologique.

5.1.2. Environnement de travail pour le traducteur

L'intégration du système Logos doté des outils de Machine Translation et de l'environnement de travail Optimizer d'Eurolang donne une idée de ce qu'il est possible de réaliser à l'heure actuelle en matière d'environnement de post-édition.

La démarche suivie est typiquement la suivante :

[5] Ce procédé est qualifié de primaire car il s'agit uniquement du stockage pur et simple des propositions et de leur traduction. Aucune règle n'est déduite quant à la manière de traduire une proposition qui n'aurait pas déjà été rencontrée (si ce n'est des formes fléchies de propositions déjà traduites).

L'utilisateur importe un texte à traduire dans son environnement de travail habituel (Word pour Windows, WordPerfect, ...) et, à partir de là, choisit les bases de données de phrases préalablement traduites ainsi que les glossaires contenant les termes et les expressions auxquels il a souvent recours. Ces bases de données et glossaires sont classés par paire de langue et par sujet. L'idée est de mettre à disposition de l'utilisateur toutes les informations dont il est susceptible d'avoir besoin à un moment ou un autre. Pour cela, le système stocke ces informations au fur et mesure dans un fichier temporaire appelé *folder*. Les phrases à traduire sont comparées aux bases de données de phrases préalablement traduites. Les traductions des *perfect matches* et des *fuzzy matches* sont sauvées dans le *folder*. Les mots des nouvelles phrases et des *fuzzy matches* sont ensuite recherchés dans les glossaires et les traductions sont sauvées dans le *folder*. Les nouvelles phrases et des *fuzzy matches* sont finalement traduit par Logos et leurs traductions sont également sauvées dans le *folder*. Dans le *folder*, toutes ces données sont reliées aux phrases du texte source auxquelles elles se rapportent.

L'utilisateur passe alors à la phase de traduction (humaine) proprement dite. L'utilisateur a devant les yeux une copie de son texte source. Les phrases qui ont fait un *perfect match* sont d'une certaine couleur. Les *fuzzy matches* sont d'une autre couleur et les propositions qui ont été trouvées dans le glossaire et qui se situent à l'intérieur d'un *fuzzy match* ou d'un non-*match* sont d'une couleur encore différente. Lorsque l'utilisateur clique sur un *perfect match*, sa traduction est chargée à partir du *folder* et vient remplacer la phrase source. Pour les *fuzzy matches* l'utilisateur regarde la traduction qui est proposée et la modifie. Un fois cela fait, la traduction remplace la phrase source. Enfin, lorsqu'il traduit une phrase dont une proposition a été rencontrée dans le glossaire, en cliquant sur la proposition, l'utilisateur visionne la traduction et un clic supplémentaire insère celle-ci dans la traduction. Au fur et à mesure de la traduction, les phrases traduites et les ajouts de terminologies sont stockés respectivement dans la base de données de phrases et dans le glossaire.

5.2. Translatability Index

5.2.1. Présentation

Le *Translatability Index* (TI) est une valeur qui représente la difficulté d'un texte à être traduit. Pratiquement, son but est d'évaluer la qualité qu'aura la traduction finale d'un texte dès la fin du module *Res*. Le premier avantage d'un tel outil est qu'il permettrait d'économiser du temps et des ressources en ne soumettant pas des textes dont la traduction ne pourrait pas être satisfaisante. Ensuite, l'identification des facteurs qui rendent la traduction difficile

donnerait la possibilité de guider l'utilisateur dans la correction du texte source (voire même dans sa rédaction initiale) afin de diminuer sa complexité.

La quantification de la qualité de la traduction se fait sur une échelle allant de 1 à 7. Sur le tableau 3.8., on peut voir la correspondance entre ces valeurs et leur signification.

Translatability Index	Signification
7	Aucune modification n'est nécessaire
6	Il y a une ou deux modifications mineures à effectuer mais aucune référence à la source n'est nécessaire pour la compréhension
5	Plusieurs modifications mineures à effectuer, certaines références à la source peuvent s'avérer nécessaires pour une bonne compréhension
4	Plusieurs modifications mineures à effectuer, il est nécessaire de faire référence à la source pour les corriger car plusieurs solutions sont possibles
3	Des modifications majeures sont à effectuer
2	De grandes parties de la phrase doivent être entièrement retraduites
1	La phrase entière doit être retraduite

Tableau 3.8. : Les significations des *translatability index* (extrait de [GDANIEC_a])

La traduction d'une phrase est considérée comme insuffisante si elle obtient un score inférieur à 4 car, dans ce cas, elle ne peut pas être utilisée par la fonction de post-édition. Ces notes sont bien sûr très subjectives et on verra que cela pose beaucoup de problèmes, principalement dans l'approche mathématique de l'étude de ces valeurs. Mais elles sont nécessaires à l'évaluation des phrases par des linguistes externes à l'entreprise LOGOS pour qu'il y ait une certaine harmonie entre leurs estimations. En effet, le but du projet de création d'un *translatability Index* est de trouver une méthode qui permette de calculer automatiquement, et pour chaque phrase, la valeur du TI qui se rapproche le plus possible de celle donnée par des évaluateurs humains qui sont appelées *Quality Index* (QI).

5.2.2. Principe

Les méthodes qui ont été mises au point se basent sur l'existence de corrélations entre certaines propriétés d'un texte et sa complexité. Par exemple, le tableau 3.9. montre les résultats obtenus lorsqu'on compare la longueur moyenne de phrases écrites en anglais et leur *Quality Index* (QI) en allemand qui correspond à la qualité de la traduction

Quality Index	Longueur moyenne des	Nombre de phrases

	phrases	
7	15	93
6	21	126
5	23	163
4	24	134
3	26	67
2	31	14

Tableau 3.9. : Rapport entre le *quality index* et la longueur des phrases. On remarque que la longueur moyenne des phrases a une relation linéaire avec la qualité de la traduction (extrait de [GDANIEC_a]).

Le même test a été réalisé pour d'autres paires de langues (les résultats les plus intéressants sont regroupés dans les rapport [GDANIEC_a] et [GDANIEC_b]). Les résultats diffèrent bien sûr en fonction de la langue source puisqu'elles comportent chacune des difficultés qui leur sont propres (les verbes sous forme progressive - qui se terminent par « ing » - en anglais par exemple). Mais les corrélations sont également variables pour des langues cibles différentes alors que la langue source est la même. On remarque ce phénomène lorsque, par exemple, les limites d'une proposition relative provenant d'une phrase écrite en anglais ne sont pas correctement repérées par le système ; cela n'influencera pas la traduction en français car l'ordre des mots est le même, mais en ce qui concerne la traduction en allemand, le verbe ne se trouvera pas à la bonne place.

D'autres propriétés ayant une corrélation avec la qualité de la traduction ont été identifiées. Ces propriétés peuvent être grammaticales (présence d'une proposition relative) ou syntaxique (nombre d'homographes), mais elles peuvent aussi dépendre du système (le mot ne se trouve pas dans le dictionnaire). Leur identification s'est principalement déroulée de manière empirique en cherchant les sources des erreurs de traduction. Cette manière de procéder explique leur nombre élevé et leur caractère parfois très spécifique. Le tableau 3.10. montre ces différentes propriétés qui sont divisées en 3 catégories : les propriétés communes à l'anglais et à l'allemand, les propriétés spécifiques à l'anglais et les propriétés spécifiques à l'allemand.

Propriétés communes à l'anglais et à l'allemand
Longueur de la phrase
Nombre de mots qui n'ont pas été trouvés
Phrase coupée parce qu'elle est trop longue
Nombre de parenthèses courtes
Nombre de parenthèses longues
Nombre de conjonction de coordination
Absence de verbe

Toutes les lettres sont écrite en majuscule
Toutes les initiales des mots sont écrites en majuscule
La phrase est interrogative
Nombre d'éléments qui se trouvent avant le verbe
Le verbe est entre guillemets
Les parenthèses ne correspondent pas
Nombre de propositions dépendantes
Nombre de parenthèses emboîtées
Nombre de propositions relatives
Nombre de virgules
Propriétés spécifiques à l'anglais
Absence de proposition principale
Nombre de propositions composées
Nombre de noms se terminant par « ing »
Nombre de clauses introduite par un verbe se terminant par « ing »
Nombre d'occurrence du mot « as »
Nombre d'occurrence du mot « as » en début de phrase
Nombre d'occurrence du mot « with »
Nombre de groupes nominaux comportant au moins 3 éléments
Nombre de groupes nom. comportant au moins 3 élém. et un mot en « ing »
Propriétés spécifiques à l'allemand
Nombre de noms ambigus
Nombre de verbes ambigus
Présence d'inversions au début de la phrase
Nombre de participes emboîtés
Nombre de pronoms
Nombre d'occurrence des mots « weder noch »
Nombre d'occurrence des mots « allein, lediglich, gerade, wer, ja, wenig, ... »

Tableau 3.10. : Propriétés qui influencent la qualité de la traduction.

Toutes ces informations sont fournies par le module *Res*. Le problème est de déterminer dans quelles proportions chacun de ces facteurs influence la mauvaise qualité de la traduction. La première méthode utilisée dans le but d'estimer ces poids suivait une logique empirique. Les résultats encourageants obtenus par cette approche ont mené à une méthode mathématique.

5.2.3. Méthode empirique

Le premier programme qui ait été réalisé pour ce projet a pour but de créer un fichier contenant, pour chaque phrase, la comptabilisation des occurrences des facteurs cités dans le tableau 3.10. Ce programme est lancé à la fin de *Res* puisque beaucoup de ces propriétés nécessitent la connaissance des rôles syntaxiques des différents éléments de la phrase.

Le second programme utilise les données du premier pour calculer le TI de chaque phrase. Les calculs s'effectuent de la manière suivante : le score de la phrase est initialisé à 7 et tous les facteurs présents engendrent une pénalisation du score. L'importance de cette pénalisation est propre à chaque facteur et est déterminée par l'utilisateur qui, pour ce faire, doit modifier le corps du programme. Le TI du texte entier est ensuite calculé en faisant la moyenne des TI de l'ensemble des phrases.

Exemple

Prenons une phrase sans verbe contenant 5 éléments. Si l'utilisateur a défini que le poids par élément de la phrase est égal à 0.05 et que le poids pour une phrase sans verbe est de 0.2, on aura le calcul suivant :

TI = 7 - (5*0.05) - 0.2 = 6.55

La démarche suivie pour établir le poids de chaque facteur afin de maximiser la corrélation entre les TI calculés et les QI estimés par les linguistes est entièrement empirique. C'est en effectuant de manière répétitive des ajustements guidés par la logique et l'intuition sur les pénalisations et les facteurs que les premiers résultats ont été obtenus. Le tableau 3.11. donne les corrélations calculées pour des textes anglais traduits en allemand.

Type de document	Nombre de phrases	QI	TI	Corrélation entre TI et QI
LEGAL	21	4.47	4.80	93.1 %
ELECTR.	32	5.87	5.94	98.8 %
BANQUE	105	4.38	4.69	93.4 %
INFORMAT.	69	4.95	5.12	96.6 %
MEDECINE	48	5.04	5.42	92.8 %
CHIMIE	79	5.05	5.00	99.0 %

Tableau 3.11. : Corrélations calculées pour des textes anglais traduits en allemand.

Ces résultats obtenus de manière très informelle laissent entrevoir de grandes possibilités d'une méthode basée sur des concepts mathématiques.

5.2.4. Méthode mathématique

C'est ce troisième programme qui a constitué notre tâche principale lors de notre stage au sein de l'entreprise LOGOS. Son but est de calculer, à partir des données toujours reçues du premier programme, la pénalisation de chaque facteur qui minimise la différence entre la qualité estimée par calcul (TI) et la qualité estimée par les linguistes (QI). Cela signifie qu'il calcule le poids que représente, en moyenne, chaque facteur dans la complexité de la phrase. Plusieurs exemples de résultats sont à la disposition du lecteur en annexe.

Ce programme utilise 3 fichiers en plus de celui qui contient les informations sur le nombre d'occurrences de chaque facteur pour chaque phrase. Le premier contient la liste des textes sur lequel le programme se base pour effectuer les calculs. Le second contient les QI donnés par les linguistes pour ces textes et le troisième permet à l'utilisateur de sélectionner les facteurs dont le programme doit tenir compte pour effectuer ses calculs.

Un double filtrage est effectué dans le but de minimiser les incohérence et les redondances mais aussi d'augmenter la flexibilité. Ce sont les phrases qui sont filtrées en premier lieu. L'utilisateur peut déterminer les phrases qui ne doivent pas être retenues par le programme en leur donnant un QI égal à 0. Ensuite, le programme ne tient pas compte d'une phrase ayant exactement les mêmes caractéristiques (le même nombre d'occurrence de chaque facteur) qu'une autre. Le second filtrage est réalisé au niveau des facteurs. Comme nous venons de le voir dans le paragraphe précédent, l'utilisateur a la possibilité de choisir les facteurs qui doivent être pris en compte. Ensuite, le programme filtre les facteurs qui n'apparaissent jamais dans le(s) texte(s) analysé(s) ainsi que ceux qui ont toujours la même valeur (ce dernier test a été rendu obligatoire par la seule présence d'un facteur indiquant le langage source utilisé). De plus, l'optimisation effectuée par le programme nécessite l'inversion d'une matrice. Pour pouvoir effectuer cette opération, il est impératif que les vecteurs de la matrice soit indépendants. Si ce n'est pas le cas, les facteurs correspondant aux vecteurs gênants sont supprimés.

Les poids calculés sont placés dans un fichier qui sera utilisé par d'autres programmes annexes qui sont employés pour l'interprétation des résultats. Sur base de ces poids, la différence moyenne entre les TI et les QI est donnée pour l'ensemble des phrases mais aussi pour chaque ensemble de phrases ayant des valeurs identiques pour le QI. Enfin, un test statistique est effectué afin de déterminer le taux de corrélation de chaque facteur par rapport aux QI et le programme indique, dans l'ordre décroissant, le résultat de ce test ainsi que l'ensemble idéal de facteurs à considérer.

5.2.5. Conclusion

Les résultats obtenus sont loin de répondre aux attentes. Différentes hypothèses ont été testées avec la méthode mathématique :

- même nombre de phrase pour chaque QI
- filtrage des phrases ayant un QI supérieur ou inférieur à certaines valeurs
- sélection de différents ensembles de facteurs
- ...

Dans tous les cas de figure, les résultats manquent de régularité. Lorsque l'on utilise les poids calculés pour déterminer les TI d'autres textes, on constate que, dans la majorité des cas, on s'approche de manière très satisfaisante des QI donnés par les linguistes. Malheureusement, il reste toujours quelques textes dont le résultat est très décevant. Un outil basé sur cette technique ne pourrait donc pas faire preuve de la fiabilité minimum qui serait nécessaire pour qu'il soit mis à la disposition des clients.

Ce manque de régularité entre les différents textes peut être expliqué par le fait que les QI sont donnés par différents linguistes. Les critères qui leur sont fournis sont loin d'être suffisamment objectifs (voir tableau 3.8.). La conséquence est que le programme doit attribuer les poids en tenant compte de différentes subjectivités. Ce n'est bien sûr pas la seule explication à l'échec de cette méthode. Même si les QI étaient attribués par un seul linguiste, il serait difficile pour lui de le faire de manière totalement cohérente, et surtout, les différents facteurs n'ont pas toujours la même influence négative sur la traduction.

Les bons résultats obtenus avec la première méthode peuvent être expliqués par le fait que c'est sur base de l'étude de leur traduction que les critères ont été identifiés empiriquement et que, ensuite, les poids ont été estimés. Des tests ont également été réalisés sur des nouveaux textes mais leur nombre trop restreint ne permet pas d'en tirer une conclusion.

Malgré les résultats décevants du calcul des TI, l'identification des facteurs qui rendent la traduction difficile peut s'avérer utile. Des expériences ont montré qu'en réécrivant des textes, les QI donnés par les linguistes augmentaient de manière significative.

Cette conclusion n'est que provisoire car le projet n'a pas été abandonné. Les recherches continuent sur la manière d'utiliser éventuellement les résultats actuels et d'autres approches font également l'objet de recherches.

5.3. Les Semcodes

Bien que les *SAL codes* constituent une aide très précieuse dans le cadre de la résolution des ambiguïtés, l'analyse des traductions met en évidence la faiblesse de ces codes à 2 points de vue : les niveaux d'abstraction et le nombre de significations encodables. Les 3 niveaux (*superset*, *set* et *subset*) dont disposent les *SAL codes* ne permettent pas de différencier de manière satisfaisante les aspects syntaxiques et sémantiques. Cela entraîne le regroupement dans une même catégorie de mots nécessitant, dans certains cas, des traitements distincts. Mais le désavantage principal des *SAL codes* est qu'il n'autorise la prise en compte que d'un aspect syntaxique et/ou sémantique puisque chaque élément dispose d'un seul code par *part-of-speech* possible.

Exemple

Le mot « table » en tant que nom peut être utilisé comme :

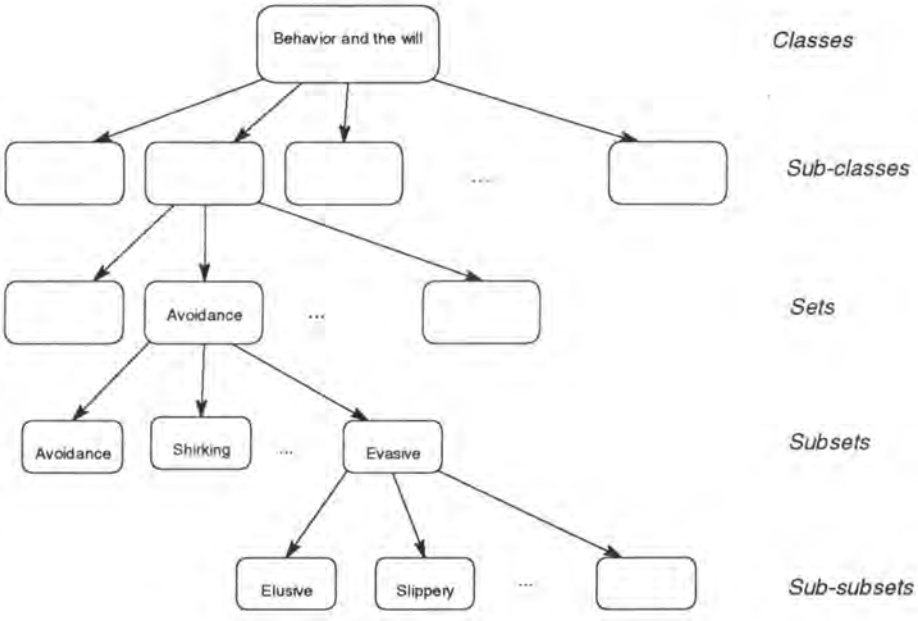
- surface représentant un support.
- une information (table d'un document).

c'est pour combler ces lacunes que le projet de créer une seconde taxonomie plus complète appelée *Semcodes* a été lancé.

5.3.1. Définition

Les *Semcodes* représentent une taxonomie en 5 niveaux de généralité, basée sur celle proposée par Roget. Au niveau le plus haut, il existe 15 *classes* qui sont subdivisées en *sub-classes*, *sets*, *subsets* et *sub-subsets*.

Exemple



Les 2 niveaux supplémentaires par rapport aux *SAL codes* permettent de mieux gérer la richesse du langage naturel, surtout grâce au fait qu'une même *part-of-speech* pourra disposer de plusieurs *Semcodes*. De cette manière, le système pourra tenir compte de toutes les significations possibles de cet élément.

5.3.2. Utilisations

Le but premier de la création des *Semcodes* était de pallier les lacunes des *SAL codes*. Nous allons donc montrer comment les *Semcodes* permettraient de résoudre des ambiguïtés qu'il n'est pas possible de traiter actuellement avec les seuls *SAL codes* au niveau de *Res*, des *Trans* et de SEMTAB.

A) Au niveau de Res

Dans une phrase telle que : « The DB manager prepares objectives and plans for the forthcoming development cycle. », le système n'est pas en mesure de déterminer si le mot « plans » est utilisé en tant que nom ou verbe. La règle qui permettrait de résoudre ce problème n'est pas appliquée car elle dispose d'une condition qui précise que les éléments « objectives » et « plans » doivent être liés sémantiquement. La vérification de cette condition est effectuée par la comparaison des *SAL codes* de ces éléments, or ceux-ci n'ont aucun point commun. La précision supplémentaire offerte par les Semcodes permettrait de mettre en évidence les relations qui existent entre ces 2 mots.

B) Au niveau des Trans

En présence d'une phrase contenant les mots : « old people and children », l'analyse de la structure entraînerait que l'adjectif « old » soit appliqué aux 2 éléments « people » et « children » car ils sont tous les 2 de type humain. Les différents Semcodes correspondants à ces éléments seraient ceux présentés au tableau 3.12.

Semcodes de « men »	Semcodes de « children »
...18.9 workforce	...302.2 young people
...76.3 mankind	.559.5 family
...577.11 staff	

Tableau 3.12. : Semcodes des éléments « men » et « children ».

On n'observe aucune relation sémantique qui justifierait l'application de l'adjectif « old » au mot « children ». Ce changement ne nuirait pas à l'application de la règle dans le cas où l'adjectif s'applique effectivement aux éléments qui suivent tel que pour : « smart boys and girls ». On observe sur le tableau 3.13. qu'il existe une relation suffisante au niveau des codes 302.5 et 302.4 (qui sont en gras) pour que la règle soit effectivement appliquée.

Semcodes de « boys »	Semcodes de « girls »
... 87.8 heroin	...77.5 woman
...302.5 lad	...87.6 cocaine
...577.11 man	..104.15 ladylove
...758.2 card	**...302.4 girlie**
	...577.8 maid

Tableau 3.13. : Semcodes des éléments « boys » et « girls ».

c) Au niveau de Semtab

Le niveau plus élevé de détail des Semcodes permettrait d'augmenter la précision dans le choix des traductions contextuelles, ce qui représente la tâche principale de *Semtab*.

L'existence de ces codes a entraîné l'étude d'autres utilisations possibles qui pourraient favoriser la résolution de certains problèmes que rencontre le système. Un projet intéressant a été imaginé pour faciliter le traitement des polysèmes.

5.3.3. Le traitement des polysèmes

Le but de ce projet est de définir, parmi les différents polysèmes possibles pour chaque élément, celui qui correspond à l'utilisation faite dans une phase donnée. Chaque polysème étant représenté par un *Semcode* distinct, cela reviendrait à déterminer le Semcode qui convient au contexte de la phrase.

Le principe de base est de chercher une correspondance parmi tous les Semcodes de l'ensemble des éléments qui constituent la phrase. La figure 3.4. illustre ce principe (les codes attribués ne correspondent aucunement à ceux qui seraient réellement utilisés par Logos, leur seul but étant de faciliter l'explication du principe) pour la phrase suivante : *"The band played a piece composed by Elgar"*.

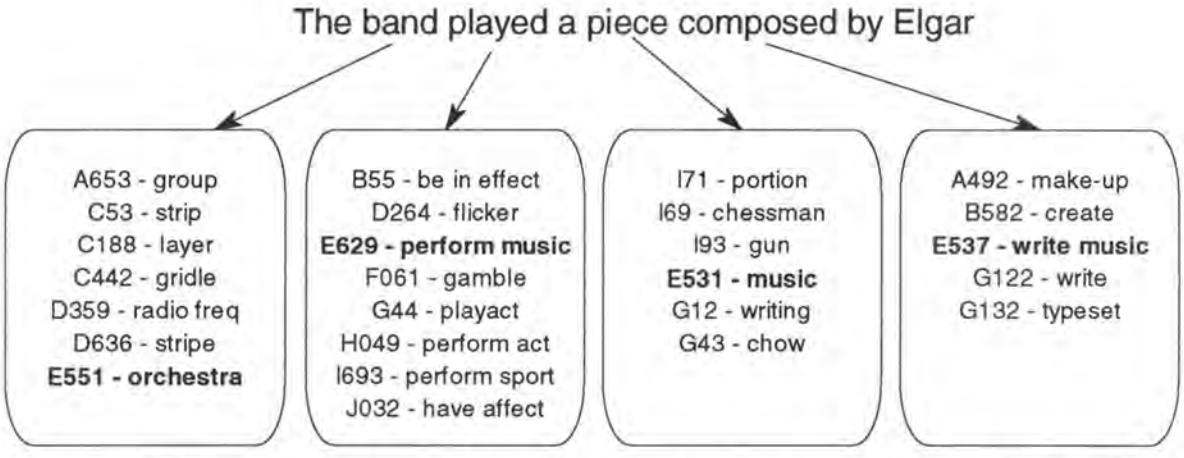


Figure 3.4. : Le code représenté par la lettre « E » est le seul qui soit commun à tous les éléments de la phrase.

C'est en se basant sur ce principe qu'un algorithme de recherche des similitudes entre les différents Semcodes auxquels les différents éléments de la phrase font référence a été imaginé. Il s'agit d'identifier, dans un espace à 5 dimensions (un par niveau de Semcodes), les codes qui sont les plus souvent rencontrés. L'exemple ci-dessous montre comment la recherche serait effectuée dans le cas où il n'y aurait que 2 niveaux de code :

Exemple

« The agency will provide both room and board. Board will consist of two nourishing meals a day, served at noon and 5 pm. ».

La distribution des SEMCODES peut être représentée sur une grille ayant 8 colonnes correspondant au niveau d'abstraction le plus élevé et 10 lignes correspondant au second niveau. Le résultat serait le suivant :

codes	1	2	3	4	5	6	7	8
1		...		.			.....	
2		....					.	
3						.	...	.
4		.....					.....	
5								
6	.		.				....	
7	.							
8								
9								
10	..							

On observe la concentration la plus élevée au niveau de la case faisant référence aux codes 2 et 4. Ce sont donc les polysèmes les plus proches de cette région de la grille qui seront sélectionnés.

Cette méthode nécessite encore beaucoup d'études car il existe certaines phrases qui pourraient provoquer des décisions erronées.

Exemple

« He sat by the sea and listen to the sound of the waves »

Dans cette phrase, le résultat de l'algorithme pourrait être faux car le mot « sound » serait interprété dans le sens « bras de mer » au lieu de « son ».

Malgré tout, le principe semble très intéressant car un tel outil représenterait à coup sûr une aide très précieuse pour le système. Il faut noter que c'est la première fois que la connaissance linguistique se trouve dans l'algorithme lui-même, bien que ces connaissances soient surtout basées sur des probabilités concernant la tendance qu'ont certains mots à se trouver dans les mêmes phrases.

CHAPITRE IV : Critique du Système

Dans cette section, nous allons d'abord situer le système Logos dans le domaine de la TA (pour cela, nous ferons référence au chapitre II qui a présenté le cadre général de ce domaine). Ensuite, nous montrerons les faiblesses et les atouts du système Logos du point de vue linguistique et informatique. Enfin, avant de conclure, nous décrirons l'environnement de travail des linguistes qui constitue un indice concernant les perspectives d'avenir du système.

1. STRATEGIE DE BASE DE LOGOS

Les modèles linguistiques vont des approches directes jusqu'aux approches dites d'*Interligua*. Le lien entre les 2 modèles est représenté par les approches transferts dont le niveau est proportionnel à l'importance du module bilingue du système.

Il est évident que le système Logos est de type indirect puisque le résultat final est loin de correspondre à une traduction "mot-à-mot" du texte source. Par contre, on pourrait le considérer, à première vue, comme un système *Interligua*. En effet, Logos dispose d'une représentation interne dans laquelle n'importe quelle langue pourrait être traduite et sur base de laquelle n'importe quelle langue pourrait être générée. Ce langage serait composé de tous les codes numériques (*word class, SAL code, form* et *overflow*) qui sont attachés à chaque élément de la phrase ainsi que les codes qui se trouvent dans les tableaux CELL et SCON. Dans ce cas, les modules Gerdem et *Res* seraient considérés comme l'étape d'analyse, et les modules *Trans* et Fprint comme l'étape de génération. Cependant, il ne peut s'agir d'un système de type Interlingua car les règles des modules *Trans* sont écrites en fonction de la langue source. On peut le remarquer à 2 niveaux :

- Le système profite de similitudes qui existent entre la langue source et la langue cible pour ne pas traiter certaines caractéristiques syntaxiques et/ou sémantiques du texte, la traduction par défaut étant de toute façon correcte.

- Les règles des *Trans* disposent de fonctions agissant sur la source

Le système Logos est donc de type transfert et il émerge rapidement que le niveau du transfert est élevé (voir point 3.3 du chapitre II). Comme nous allons le montrer, les tâches effectuées par le module bilingue sont très importante, mais identifions d'abord les 3 étapes qui constituent l'approche transfert (la figure 3.1. présentée dans la section 2 décrivant l'architecture du système illustre parfaitement ce propos). Les modules Preproc , Gerdem et *Res* sont invariants pour une même langue source, ils constituent donc l'étape d'analyse. La seule objection à cette affirmation concerne la présence de la traduction par défaut dans le dictionnaire bilingue mais on peut considérer qu'il s'agit uniquement d'une optimisation du nombre d'accès au dictionnaire. En effet, il suffirait d'accéder à un dictionnaire réservé pour la langue source lors de Gerdem et de disposer d'un dictionnaire comportant tous les mots de la langue cible au lieu d'utiliser un dictionnaire bilingue contenant les traductions par défaut (qui sont d'ailleurs séparées au niveau physique des informations concernant la source) et d'un dictionnaire cible contenant les autres traductions possibles. L'étape bilingue du transfert est évidemment représentée par les modules *Trans* et l'étape de génération par le module Fprint puisqu'il est le seul à être invariant pour une même langue cible (excepté Preproc mais ce module n'a qu'un rôle de formatage).

L'approche transfert de Logos est de niveau élevé car les modules *Trans* s'occupent presque entièrement de la génération (Fprint n'effectuant que 4 opérations linguistiques de base). De plus, nous avons montré que les *Trans* avaient pour rôle de gérer les erreurs commises par *Res* en raison du nombre de POS dans le dictionnaire bilingue. Ces tâches qui devraient normalement être dévolues aux modules d'analyse leur sont donc également attribuées.

Ce niveau élevé est un frein à l'évolution de Logos vers un système multilingue. Si la séparation entre algorithme et donnée est claire (voir description de l'architecture à la figure 3.1.), les modules pouvant être réutilisés pour la création de nouvelles paires de langues ne sont pas suffisamment importants, surtout en ce qui concerne la langue cible. Les règles des *Trans* doivent être réécrites pour toutes les nouvelles paires de langues et c'est ce qui constitue la plus grande partie du processus de traduction.

Enfin, le système Logos est de type non-interventionniste car son objectif est de maximiser la qualité de la traduction sans intervention d'un opérateur humain lors du processus de traduction. C'est à la suite de ce processus que l'intervention des linguistes est nécessaire afin de rendre la traduction correcte lors de la post-édition.

Il faut noter que le processus de traduction peut être guidé par l'intermédiaire des *subject matter codes* et des *company codes* qui précèdent le lancement du processus. Ce sont les

seules connaissances pragmatiques (voir points 3.4 et 3.5 du chapitre I) utilisées dans le processus de traduction.

2. *CRITIQUE DU POINT DE VUE LINGUISTIQUE*

Les règles utilisées, aussi bien pour *Res* que pour les *Trans*, ne sont basées sur aucune théorie linguistique générale mais sur des principes reconnus de diverses grammaires. Ces principes peuvent être rapprochés des grammaires de valence (voir point 3.4 du chapitre I) et des grammaires de cas (voir point 3.5 du chapitre I) en raison de la présence des *SAL codes* qui se prêtent bien à ce genre de théorie. Ensuite, c'est selon une méthode principalement empirique que des règles ont été ajoutées dans la base de données. C'est en se basant sur les traductions et en tentant de trouver des points communs entre les différentes erreurs décelées que les linguistes construisent leurs règles.

Cette manière de travailler entraîne la multiplication de règles utilisant des astuces permettant de traiter certains cas qui n'étaient pas prévus par le système. Il en résulte une complexité très importante des règles qui, par conséquent, ne sont accessibles qu'au seul linguiste qui les a écrites, même si la politique de Logos est de disposer de règles très simples. Ce problème peut être, en partie, attribué au fait que le langage utilisé par les linguistes pour l'écriture des règles soit de très bas niveau (nous reviendrons sur ce problème dans la section 4). En effet, il leur est très facile de détourner les tableaux et les fonctions de leur utilisation standard dans un seul but de résultat. De plus, la représentation interne de la phrase n'est pas homogène durant tout le processus de traduction. Pour s'en convaincre, il suffit d'observer le tableau CELL qui est utilisé dans des rôles différents selon qu'on se trouve dans le module Res ou dans les modules Trans.

Ce manque d'accessibilité aux règles est accru par le fait que la documentation fait cruellement défaut. La nécessité d'avoir de bons résultats le plus rapidement possible ne laisse pas de place pour la tenue à jour d'une documentation claire et détaillée.

Le très grand nombre de règles place le système Logos dans une position où les linguistes ne maîtrisent plus entièrement la situation. Des tests ont été effectués sur le module Res (pour l'anglais) afin de déterminer, sur un ensemble très important de textes, les règles qui sont le plus souvent appliquées. Il est apparu qu'un tiers de celles-ci (ce qui représente plus de 4000 règles) ne sont jamais utilisées. Mais d'autres tests basés sur des textes différents ont montré

que quelques règles inutilisées précédemment sont malgré tout parfois nécessaires. Par conséquent, il serait trop risqué de tenter de diminuer la redondance et la complexité de l'ensemble des règles.

Nous allons maintenant mettre en évidence les faiblesses du système Logos à différents niveaux : la représentation de la relation entre syntaxe et sémantique, le traitement des expressions, les référence d'un niveau supérieur à une seule phrase, les ellipses et les polysèmes.

2.2. Syntaxe et sémantique

L'utilisation simultanée de connaissances syntaxiques et sémantiques est bien sûr représentée par les *SAL codes*. Bien que constituant un des points forts du système Logos, il existe encore beaucoup de cas où cette taxonomie est prise en défaut.

Exemples

La phrase : "They brought fresh fruits and vegetables from the farm" sera traduite par : "ils ont apporté des fruits frais et des légumes de la ferme". L'adjectif "frais" aurait dû s'appliquer également au mot "légumes" mais celui-ci ne dispose pas de code commun avec le mot "fruit".

Le problème peut se poser dans l'autre sens, la phrase : "they love the blue sky and the sunshine" sera traduite par : "ils aiment le ciel et le soleil bleu" car, dans ce cas, les mots "sky" et "sunshine" ont des codes sémantiques communs.

La solution proposée par Logos est la création d'une taxonomie comportant 2 niveaux d'abstraction supplémentaires et autorisant la prise en compte de plusieurs codes pour un même élément. Ces nouveaux codes, appelés Semcodes, ont déjà été présentés à la section 5.2.

2.3. Traitement des expressions idiomatiques

Nous avons expliqué que les responsables du système Logos ont décidé d'insérer des entrées contenant plusieurs mots dans le dictionnaire et de les traiter comme un seul élément par la suite. Ce choix est motivé par l'existence d'expressions idiomatiques dont le sens ne peut pas être déduit de la somme des significations des éléments qui la composent.

Il existe cependant un problème qui est que le processus d'analyse de la phrase pourrait se voir cacher des informations sur la structure de celle-ci si un groupe de mots en vient à être traité comme un mot. C'est principalement le cas si le groupe de mots considéré contient un élément verbal. Supposons par exemple qu'on ait placé "here is" (trad. de "voici") dans le dictionnaire et que l'on veuille traduire la phrase "The data listed here is available". L'analyse de phrase échouera puisqu'elle ne trouvera pas de verbe.

Pour éviter ce genre de problème, un principe a été établi disant qu'une telle démarche ne doit être entreprise que si la fréquence d'utilisation du groupe de mots avec la signification envisagée est extrêmement commune. Cela signifie qu'il faut que les cas où le groupe de mots obéit à une autre structure syntaxique et/ou à une autre signification soient marginaux, voire inexistants.

Pour résumer, nous pouvons donc dire que l'insertion de groupes de mots comme entrée du dictionnaire est très pratique et très efficace en ce que la traduction correcte du groupe de mot se limite à la simple consultation du dictionnaire. Mais il faut tenir compte du fait que cela peut entraver l'analyse qu'effectue le système.

2.4. Les anaphores

Le système Logos traite les différentes phrases du texte de manière indépendante. Il n'est donc pas en mesure de traiter les cas où la traduction correcte d'un mot nécessite la référence à une phrase précédente.

Exemples

Dans le texte suivant : "He bought a new car. He drives it fast.", le système ne dispose d'aucun critère pour déterminer si le mot "it" doit être traduit par "le" ou "la".

Dans cet autre texte : "this processus has a new kind of decision gate. This gate ...", l'expression "décision gate" recevra une traduction particulière mais elle ne s'appliquera pas à l'occurrence du mot "gate" dans le seconde phrase.

La gestion des anaphores est un problème typique rencontré par les systèmes de Traduction Automatique. A l'heure actuelle, aucune recherche approfondie n'a été réalisée en la matière en ce qui concerne le système Logos, ce genre de problème ne constituant pas une priorité dans les perspectives d'avenir du système.

2.5. Les polysèmes

Le fait que tous les éléments disposent d'un seul *SAL code* pour chaque *part-of-speech* empêche le système de tenir compte des différentes significations que peut avoir un mot pour une même catégorie syntaxique. La solution proposée par les responsables à déjà été présentée lors de l'explication des Semcodes à la section 5.2 du chapitre III.

3. LE POINT DE VUE INFORMATIQUE

Le système Logos étant assez âgé, l'entièreté des programmes qui constituent le système en lui-même sont écrits en FORTRAN 77. Cela signifie que même la gestion des fichiers est prise en charge par ces programmes, ce qui implique que le système ne dispose aucunement des dernières améliorations en matière d'optimisation des accès.

Cette faiblesse du système est actuellement prise en charge par la compagnie Logos. Un système disposant d'une base de données relationnelle (ORACLE) est en développement. La disponibilité d'un tel outil a ouvert de nouvelles possibilités au système Logos. Par conséquent, les responsables du projet ont décidé de modifier en partie l'architecture du système. Un des grands changements se situerait au niveau de la séparation des phrases qui serait entièrement effectuée par Gerdem et plus par Preproc.

4. L'ENVIRONNEMENT DE TRAVAIL DES LINGUISTES

La difficulté du travail des linguistes réside dans le fait qu'ils doivent écrire leurs règles dans un langage de très bas niveau (même si aucune connaissance en programmation n'est requise). Toutes ces règles étant exclusivement constituées de nombres (si on ne tient pas compte des commentaires), leur élaboration et leur compréhension à posteriori est souvent très ardue même pour le linguiste qui les a écrites.

Malgré tout, plusieurs outils ont été mis à la disposition des linguistes afin de faciliter leur tâche. La mise en place de sous-ensembles de règles propres à chaque linguiste est certainement le plus utile. Il leur donne la possibilité de constater l'effet de l'introduction de

nouvelles règles dans le système sans que celles-ci ne soient physiquement insérées dans la base de donnée. De cette manière, chaque linguiste peut procéder à ses essais sans interférer avec le travail effectué par ses collègues.

Il est facile d'imaginer une interface permettant aux linguistes de construire leurs règles sans être obligé de donner explicitement tous les codes numériques. Il suffirait de créer un logiciel qui donnerait la liste des différents types de patterns, de conditions et de *switchs*, l'utilisateur n'ayant plus qu'à choisir celui qui lui convient. Le logiciel placerait alors le code correspondant au choix de l'utilisateur dans la règle. De nouveau, bien qu'un tel programme soit très simple à réaliser, cela ne fait pas partie des projets prioritaires car le gain de productivité ne paraît pas suffisant, surtout à court terme.

Lorsqu'une règle est écrite, des procédures (appelées *checkers*) chargées de vérifier sa validité sont exécutées. Elles vérifient si le format est correct et elles passent en revue les différentes fonctions du VTR pour identifier d'éventuelles contradictions. Il faut noter qu'il existe même un *switch* permettant d'indiquer au *checkers* de ne pas tenir compte des instructions suivantes car le linguiste sait pertinemment que le système refuse ce genre de procédé. Cette technique illustre parfaitement la critique formulée au début de cette section concernant la manière de travailler de Logos qui consiste à utiliser un grand nombre d'astuces à plusieurs niveaux.

5. CONCLUSION

Comme nous venons de le voir, le système peut être amélioré à plusieurs points de vue. Malgré cela, il faut remarquer que les résultats obtenus jusqu'à présent sont à classer parmi les meilleurs en ce qui concerne les systèmes *general purpose*. La figure 4.1 décrit la progression réalisée au niveau de l'ensemble des paires de langue jusqu'en 1990.

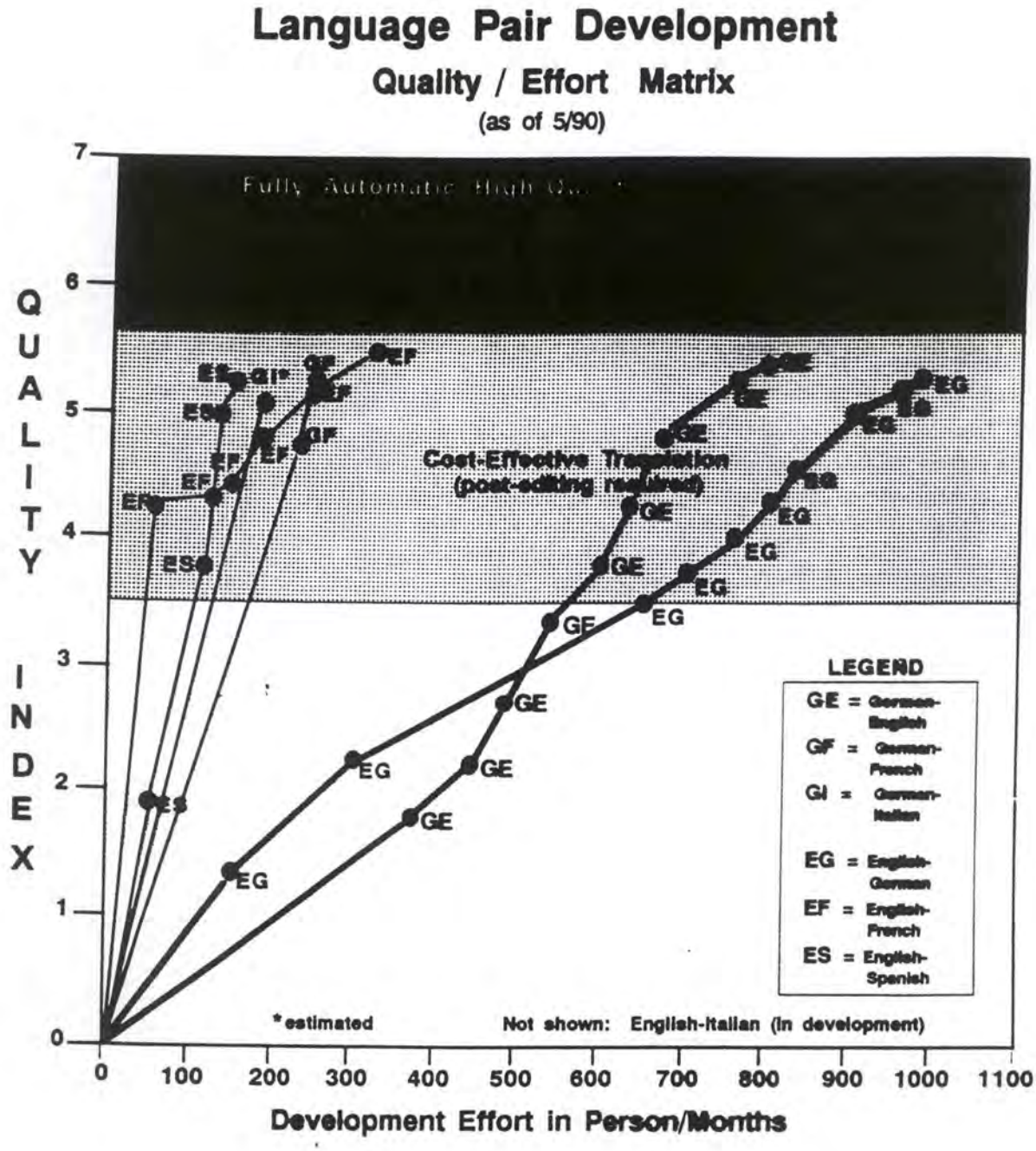


Figure 4.1. : Progression estimée par des linguistes externes à l'entreprise Logos.

Les systèmes traduisant de l'anglais vers l'allemand et inversement étant les premiers mis au point, on constate que l'introduction de nouvelles paires de langues est grandement facilitée par les travaux précédemment réalisés sur les autres langages. Cette constatation permet d'être optimiste quant à l'évolution du système Logos vers un système multilingue. (+ séparation algorithme - données)

Par contre, le manque de rigueur dans l'écriture des règles risque d'hypothéquer les chances de Logos de progresser encore de manière significative au sein des paires de langues déjà élaborées. En effet, même si le principe du *pattern matching* offre au système la possibilité d'ajouter un grand nombre de règles sans affecter les performances, cela augmente la complexité de l'ensemble des règles. Ce qui mènera le système Logos dans la position où l'ajout d'une règle pour corriger certaines erreurs aura des conséquences néfastes sur des traductions qui étaient correctes auparavant.

Troisième partie :

Principaux systèmes de TA

Chapitre V : Eurotra

1. INTRODUCTION

En 1976, sentant que la demande de traductions allait croissant, la Commission des Communautés Européennes adopta le système américain Systran (voir chapitre VII) mais, déçue par les limites de son multilinguisme et de la qualité de ses traductions, on décida de lancer un projet dans le but de *créer un système de TA de conception avancée et capable de traiter toutes les langues officielles de la communauté*[6]. En 1986, cela signifiait 9 langues (danois, néerlandais, anglais, français, allemand, italien, grec, portugais et espagnol) c'est-à-dire 72 paires de langues. Eurotra est de loin le plus important projet de TA qu'il y ait jamais eu. 18 institutions, situées dans tous les états membres étaient impliquées. En 1989, cela représentait 150 personnes.

Mais le projet ne livra pas les résultats escomptés et l'objectif de la création d'un système informatique fut abandonné. Malheureusement, les seuls apports d'Eurotra furent d'ordre linguistique. Ceux-ci se placent au niveau de la linguistique comparée des langues européennes et ne sont généralement pas remis en question mais il s'agit là uniquement d'un effet secondaire bien loin des ambitions initiales du projet.

Dans ce chapitre, nous allons en fait décrire les spécifications d'un système qui n'a jamais été entièrement implémenté. Les informations dont nous disposons proviennent de [MAEGAARD_a], [MAEGAARD_b] et du chapitre XIV de [HUTCHINS], qui, lui, se base principalement sur [ALLEGRANZA]. Pour plus de détails sur le projet Eurotra, le lecteur peut consulter ces références.

Ci-dessous, nous donnons d'abord un aperçu de l'architecture et du fonctionnement du système (section 2). Il nous faudra d'abord passer par quelques précisions sur l'approche linguistique d'Eurotra (section 3) avant de voir plus en profondeur comment fonctionnent les

[6]Le but n'était cependant pas de créer un système entièrement opérationnel mais plutôt un prototype qui, à la fin de sa phase de développement, pourrait donner des bases solides pour un système en "grandeur réelle".

modules d'analyse et de génération (section 4). Nous terminerons par une brève critique et une comparaison avec Logos (section 5).

2. APERÇU DE L'ARCHITECTURE ET DU FONCTIONNEMENT SYSTEME

Deux facteurs influencent principalement la façon dont est conçu les système : la répartition du travail entre différents groupes dans les états membres et la volonté de faire usage des méthodes de découpe en modules du génie logiciel afin de bénéficier d'une plus grande facilité de généraliser, d'étendre, et d'améliorer le système.

Eurotra se voulait être le premier système réellement multilingue. Dans un tel système (avec 72 paires de langues !), on penserait immédiatement à mettre en oeuvre une approche de type *interlingua*. Il aurait alors suffit à chaque groupe de travail d'un état membre donné de construire le module d'analyse et de génération pour sa langue. Mais, devant les échecs des systèmes qui tentèrent cette approche, les concepteurs décidèrent de se cantonner à l'approche *transfert*, réputée plus fiable. Cependant, pour ne pas devoir écrire 72 modules de transfert de taille importante, on décida qu'il ne s'agirait que d'un transfert au niveau lexical, c'est-à-dire qu'on n'aurait que des modules de transfert (bilingues) relativement simples établissant des correspondances entre les items lexicaux des deux langues concernées.

Au niveau le plus élevé, le système se divise en trois parties principales : l'analyse, le transfert et la génération (voir figure 5.1).

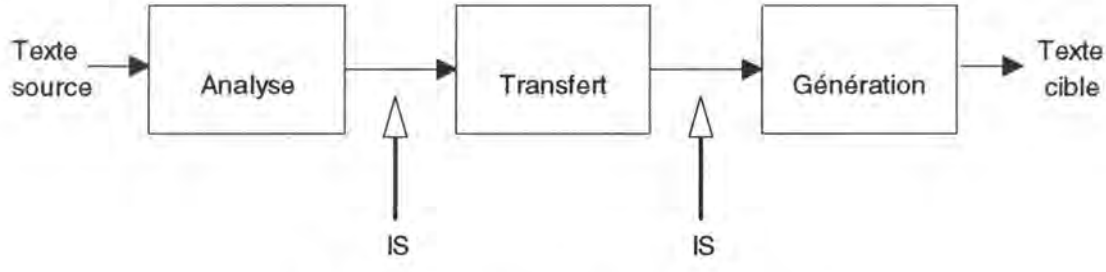


Figure 5.1. : Eurotra, vue globale du système

Alors que le module de tranfert est bilingue, les modules d'analyse et de génération sont monolingues. Ceci est important pour l'organisation des différents groupes de travail. En effet, alors qu'un groupe de travail d'un pays donné peut travailler sur les modules d'analyse et de génération de sa langue pratiquement sans concertation avec les autres, un module de transfert

doit être conçu par les groupes concernés en collaboration (nous verrons plus loin que cela doit être un peu relativisé) et c'est une autre raison pour laquelle on a essayé d'en minimiser la complexité. Les modules d'analyse et de génération s'en trouvent donc plus complexes.

Afin que les modules d'analyse (de génération) puissent être réalisés indépendamment par un groupe, il faut qu'il y ait accord sur le format de la représentation qui est fournie au (par le) module de transfert. Celle-ci est appelée *Interface Structure* (IS) et son format ainsi que le niveau d'information linguistique qu'elle contient ont du être définis préalablement. Comme on le verra plus loin, la liberté est cependant laissée aux groupes d'utiliser la stratégie linguistique de leur choix afin de produire cette IS ou de générer le texte source à partir d'elle.

A un niveau plus détaillé, la structure des modules d'analyse et de génération fait apparaître une *stratification* qui est représentée par la figure 4.2 et expliquée juste après.

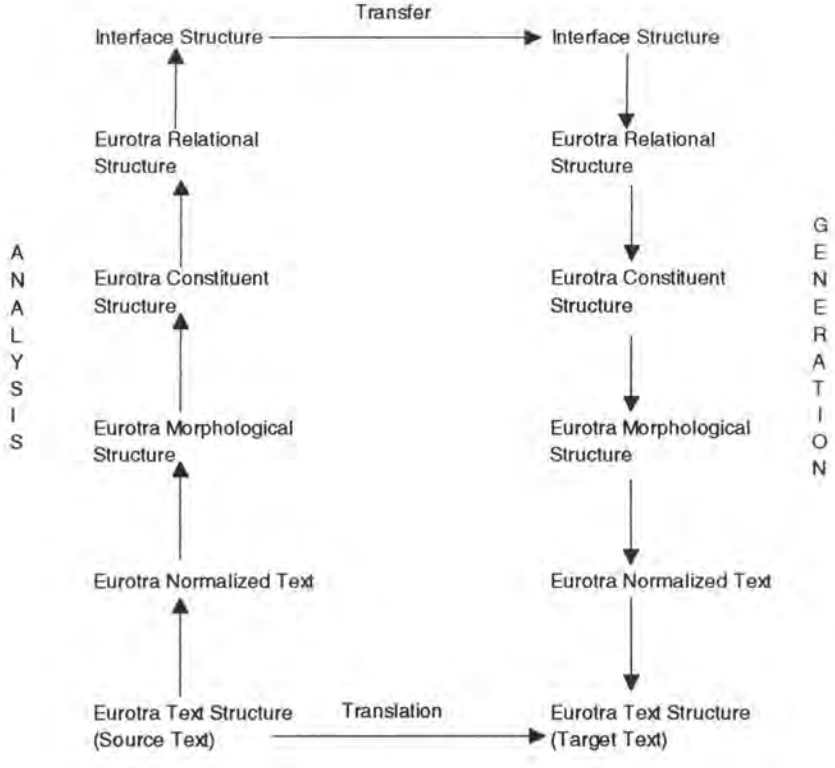


Figure 5.2. : Stratification des modules d'analyse et de génération

Les différentes étapes de ces deux modules correspondent en fait à des *mappings* entre niveaux linguistiques distincts, à savoir, morphologique, syntaxique au sens de la structure de surface, relationnel et syntaxique au sens de la structure profonde. Une étapes d'un des modules d'analyse ou de génération consiste à passer d'un niveau de représentation à un autre par l'application de règles.

3. APPROCHE LINGUISTIQUE

Les représentations du texte dans ses différents niveaux linguistiques sont réalisées dans un formalisme appelé E-framework. Il s'agit d'une grammaire d'unification (voir chapitre 1, section 3.6) où les objets de base sont les *feature bundles*, qui sont des suites de paires attribut-valeur. Pour une langue donnée, une *théorie* décrit, pour chaque niveau, les attributs autorisés, leurs valeurs possibles ainsi que les combinaisons valables de plusieurs paires attribut-valeurs dans un même *feature bundle*. Par exemple, la théorie au niveau lexical pourrait dire que (1a), ci-dessous, est valable.

Une *structure* est définie comme un ensemble de *feature bundles* sur lesquels sont appliquées des relations de *dominance* (relation père-fils) et de *précédence* (ordre sur les fils d'un même père). La *théorie* reprend aussi les structures admissibles à un niveau donné. Au niveau syntaxique (de surface), la *théorie* pourrait dire que (1a) peut se décomposer (relation de *dominance*) en (1b) et (1c).

(1a) [category=np, gender=masculine, case=nominative, number=singular]

(1b) [category=determiner, gender=masculine, case=nominative, number=singular]

(1c) [category=noun, gender=masculine, case=nominative, number=singular]

Les règles de la *théorie* sont écrites sous forme de règles d'une grammaire non-contextuelle et donc, de manière déclarative. Les objets bien formés d'après cet ensemble de règles sont appelés *objets consolidés*. Les *objets non consolidés* sont ceux qui, en cours de traitement sont lacunaires ou qui n'ont pas encore été testés "bien formés".

4. L'ANALYSE ET LA GENERATION

Comme on l'a vu à la section 2, l'analyse et la génération sont "stratifiées". Ceci implique que deux types de traitements ont lieu à l'intérieur de ce module.

- La *traduction* (terme propre au système) est le traitement qui effectue le mapping entre deux niveaux de représentation. Un *traducteur* est le traitement qui reçoit en entrée les objets consolidés d'un niveau et produit, par une analyse de bas en haut,

un ensemble de "sous-objets" non consolidés qui seront utilisés comme entrées du *générateur* du niveau suivant.

- Un *générateur* est un traitement qui consolide un objet soit en ajoutant/effaçant des relations de *dominance* entre objets, soit des relations de *précédence*.

Les *générateurs* utilisent trois types de règles :

- Les *structure building rules* : elles peuvent être soit des règles de réécriture de la grammaire non-contextuelle d'une *feature bundle* en plusieurs autres, soit des règles qui disent si une *feature bundle* particulière est valable à ce niveau.

- Les *feature rules* : elles s'appliquent aux objets consolidés en y ajoutant de l'information (ajouts d'attributs/valeurs) ou en faisant se propager de l'information à travers la structure.

- Les *filter rules* : elles s'appliquent également aux objets consolidés et elles permettent de neutraliser des structures qui ne répondent pas à certains critères. Elles sont en fait une "deuxième chance" qui est offerte aux linguistes de corriger les erreurs de "sur-génération".

Les *traducteurs* utilisent quant à eux deux types de règles :

- Les *default rules* : elles expriment les correspondances "évidentes" entre les théories des deux niveaux concernés, pour les couples attribut/valeur et les structures.

- Les *explicit t-rules* : elles expriment des correspondances non données par défaut qui peuvent contredire celles données par défaut.

Il est important de remarquer que l'IS est une représentation au niveau de la syntaxe profonde. Une représentation à un niveau inférieur aurait rendu obligatoire un transfert syntaxique (de surface), chose qu'il fallait éviter pour rendre le module de transfert aussi simple que possible. La syntaxe profonde, étant censée être commune à toutes les langues, le permet alors que la syntaxe de surface ne le permet pas. Ceci ajoute encore de la complexité au module de génération qui se voit obligé de générer une structure de surface.

C'est également pour limiter le transfert que l'on considère qu'au niveau de l'IS, toutes les ambiguïtés doivent être résolues. A nouveau, la représentation au niveau de la syntaxe profonde aide à y parvenir mais pour cela les linguistes qui écrivent les règles doivent être au courant de ce qui peut être source d'ambiguïté dans toutes les autres langues. Il convient donc relativiser le fait que l'analyse est entièrement monolingue comme nous l'avions annoncé dès le début.

4. *CRITIQUE DU SYSTEME*

L'approche linguistique se base, comme nous l'avons vu, sur un formalisme d'unification. Il n'y a cependant pas de concertation, pour l'ensemble des langues traitées sur la stratégie linguistique utilisée. Cette liberté dans la stratégie linguistique se retrouve également chez Logos.

Le principal problème d'Eurotra fut le manque de bases linguistiques théoriques au niveau de la linguistique comparée des langues européennes pour un système d'une telle étendue. Il fallu donc le créer et ce fut là l'apport majeur du projet. La recherche fut stimulée par la flexibilité et la modularité du système ainsi que par la puissance du formalisme employé qui permettent des tests faciles et rapides de règles linguistiques. Comme nous l'avons dit dans la critique de Logos, ces possibilités y sont nettement moins présentes et cela entrave une évolution plus rapide du système. Cependant la recherche n'est pas l'objectif de Logos et les bases de linguistiques comparées sont plus connues, les langues traitées étant plus "conventionnelles".

Au niveau des connaissances linguistiques utilisées dans Eurotra, un problème de taille est le traitement superficiel des aspects sémantiques. On se limite en effet à la syntaxe profonde et l'intégration d'informations sémantiques plus poussées aurait sans doute permis de résoudre certaines ambiguïtés plus facilement et notamment d'éviter les *filter rules* qui nuisent au caractère déclaratif de la grammaire. Mais on aurait alors sans doute du renoncer à la stratification du système. Les rapports qui ont conclu à l'"échec" d'Eurotra affirmaient que, peut-être, un recours à l'interactivité au lieu du traitement "batch" aurait pu suppléer le manque d'information sémantique sans forcément nuire à la stratification. Sans avoir recours à l'interactivité, Logos, par contre, utilise plus d'informations sémantiques par le biais des SAL codes. De plus, informations syntaxiques et sémantiques sont utilisées conjointement dans Logos.

Pour ce qui est de la stratégie du système, nous savons qu'Eurotra est un système de type *transfert*. La justification de cette approche n'est pas évidente à première vue quand on considère qu'Eurotra se voulait être le premier système véritablement multilingue mais les difficultés théoriques de développer une *interlingua* on fait que les concepteurs y ont renoncé. Le transfert reste cependant minimal (lexical) pour bénéficier au maximum des avantages liés à l'importance des parties monolingues par rapport aux parties bilingues. La taille du module de transfert dans Logos est beaucoup plus importante mais le nombre de langues qu'on veut traiter est nettement inférieur à celui d'Eurotra.

En ce qui concerne l'organisation des données lexicales dans Eurotra, limitons-nous à constater la particularité suivante : à chaque niveau de l'analyse et de la génération, on a des dictionnaires contenant les entrées atomiques du niveau linguistique donné (voir section 4 ci-dessus, *l-rules*). Au niveau le plus bas, ce sont des lexiques monolingues. D'autre part les deux systèmes étant tous deux de type transfert, ils disposent d'un lexique bilingue au niveau du transfert.

Enfin, au niveau de l'utilisation pratique, la pré-édition n'est requise dans aucun des deux systèmes ; par contre, la post-édition s'est révélée une nécessité.

Chapitre VI : Susy

1. *INTRODUCTION*

Susy est un système multilingue qui fut développé à l'université de Saarbrücken dans le cadre d'un projet débuté en 1967. On tenta d'abord d'adapter le système Systran à la traduction russe-allemand mais, suite à l'échec de cette tentative, on en vint à développer un nouveau prototype pour la même paire de langues. Ainsi naquit Susy (*Saarbrücker UbersetzungsSYstem*, système de traduction de Saarbrücken) qui fut ensuite étendu à l'anglais et au français bien que ce soient l'analyse et la génération de l'allemand pour lesquelles on consentit le plus d'effort. Depuis 1986, les recherches et les travaux sur Susy sont poursuivis dans la lignée du projet Eurotra (voir chapitre V). Actuellement, les travaux les plus significatifs produits à l'université de Saarbrücken se font sur le projet CAT2 qui s'inscrit également dans la lignée du projet Eurotra et au sujet duquel plus de précisions peuvent être trouvées dans [SHARP_a] et [SHARP_b].

La description de Susy qui va suivre se base sur [MAAS_a] et le chapitre XXI de [HUTCHINS] qui est consacré à ce système et qui, lui, se base principalement sur [MAAS_b]. Pour plus de précisions sur Susy, le lecteur pourra consulter ces références.

2. *APERÇU DE L'ARCHITECTURE ET DU FONCTIONNEMENT DU SYSTEME*

Susy est un système de type *transfert* et son fonctionnement peut être subdivisé en trois grandes étapes : l'analyse, le transfert et la synthèse (mot qui remplace "génération" dans le système Susy). L'analyse et la synthèse sont monolingues alors que le transfert est propre à une paire de langue.

La représentation interne des phrases à un niveau conceptuel est celle des arbres de dépendances (voir point 3.1.2 du chapitre I).

L'architecture du système présente une grande modularité et comme le montre la figure 6.1, la traduction d'un texte se fait en passant par 12 modules fortement spécialisés linguistiquement. Le résultat de l'une étant passé au suivant de manière séquentielle. Susy présente cependant la particularité de pouvoir transgresser cet ordre des opérations par des retours en arrière occasionnés par un mécanisme appelé *fail-soft RESCUE*.

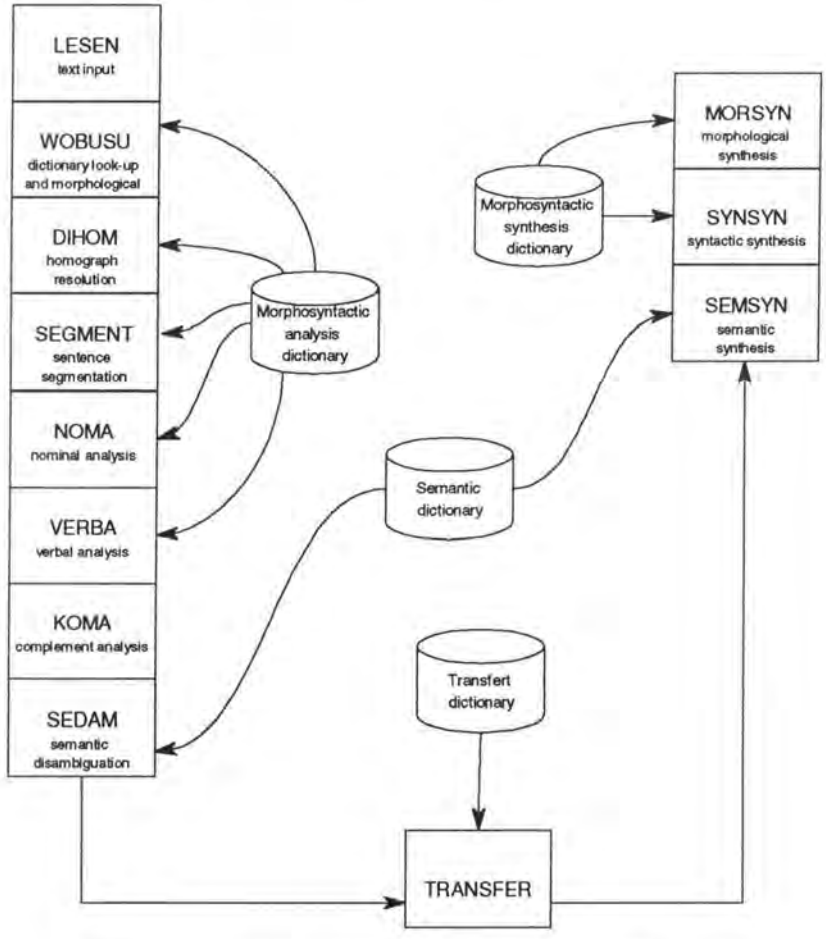


Figure 6.1 : Architecture de Susy (emprunté à [HUTCHINS])

Ci-dessous nous parlons brièvement du *fail-soft RESCUE operator* et ensuite nous détaillons le travail réalisé par chaque module.

2.1. RESCUE

Chacun des modules contient des tests de consistance de ses entrées et de ses sorties. Si, à un moment donné du cheminement de la traduction, un problème apparaît à ce niveau, on appelle la procédure *RESCUE* adéquate qui aura pour but de réexécuter les traitements du module n'ayant pas fourni les résultats escomptés en relaxant certaines trop strictes. Ce mécanisme fait que le système produira toujours une solution même si, à la limite, il s'agit d'une traduction "mot-à-mot".

2.2. L'analyse

2.2.1. LESEN - *Réception de l'entrée et identification des mots*

LESEN "lit" le texte en entrée et le divise en phrases "normalisées" c'est-à-dire dont on a séparé certaines caractéristiques typographiques (comme les majuscules) afin d'avoir une représentation homogène. On associe un *record* à chaque mot et on y inclut des informations de pré-édition éventuellement présentes dans le texte source.

2.2.2. WOBUSU - *Analyse morphologique*

WOBUSU s'occupe de la consultation du dictionnaire. En dehors des mots qui ont été marqués comme noms propres lors de la pré-édition, les mots sont recherchés dans les dictionnaires suivants :

- un dictionnaire des "mots grammaticaux" (prépositions, conjonctions, etc.)
- un dictionnaire des racines
- un dictionnaire des idiomes

Selon le dictionnaire, la correspondance est établie sur la racine ou la forme entière. Des informations d'ordre morphologique et syntaxique sont associées aux entrées et pour les noms, les verbes et les adjectifs des informations concernant les valences sont aussi incluses.

WOBUSU utilise deux procédures séquentiellement :

- La procédure *INFLECTIONS* teste d'abord s'il est possible de décomposer chaque mot en termes de *racine* + *affixe*. Par exemple, *Speichern* donnerait cours aux segmentations suivantes :

  SPEICHERN+0 SPEICHER+N SPEICHE+RN
  SPEICH+ERN SPEIC+HERN SPEI+CHERN
  SPE+ICHERN SP+EICHERN S+PEICHERN

  Ce qui donnerait naissance aux trois alternatives suivantes :

  SPEICHERN+0 *infinitif du verbe Speichern* (qui signifie *sauvegarder* en allemand)
  SPEICHER+N *datif pluriel du nom Speicher* (qui signifie *mémoire*),
  SPEICHER+N *forme fléchie du verbe Speichern*,

  dont le choix est laissé au module suivant. Dans le cas où aucune alternative n'est trouvée, on fait appel à *COMPOUND* dont le but est de traiter les mots composés (très fréquents en allemand) et qui fonctionne globalement comme *INFLECTION* mais avec des schémas plus complexes que *racine* + *affixe*.

- *IDENTITY FIXED PHRASE (IFP)* est ensuite évoquée pour repérer s'il y a dans la phrase des expressions idiomatiques. Le fait que l'analyse morphologique se fasse avant *IFP* permet d'identifier ces expressions même si certains changements (genre, temps, ...) ont eu lieu.

2.2.3. DIHOM - *"Désambiguätion" des homographes.*

DIHOM a pour but, en cas de présence d'homographes, de choisir une des alternatives fournies par *WOBUSU*. La méthode utilisée est basée sur des considérations statistiques et non sur les règles d'une grammaire. Une première routine (*SPECIAL CASES*) traite les éléments discontinus comme *werder...noch* (ni...ni) qui requièrent une stratégie spéciale. Une seconde procédure (*INHIBIT*) élimine les alternatives qui se situent dans des séquences de mots impossibles.

La dernière phase consiste à donner des poids aux ambiguïtés restantes sur base de tables de probabilités. Si plusieurs mots ambigus se suivent dans la phrase, on choisira les alternatives de

ces mots sur base de la probabilité qu'il appartiennent à telle catégorie et en tenant compte de la comptabilité des solutions pour la séquence de mots.
Notons ici l'utilisation du *fail-soft RESCUE operator*. Dans la suite du programme, si on n'arrive pas à trouver des patterns connus, on peut faire un retour en arrière à *DIHOM* et choisir l'alternative suivante dans l'ordre des homographes.

2.2.4. SEGMENT - *Segmentation des phrases en propositions*

SEGMENT identifie des éléments de la phrase qui lui permettent de délimiter les propositions principales, subordonnées, parenthésées, etc. Les éléments recherchés sont les signes de ponctuation, les conjonctions de subordination et les pronoms relatifs. Il y a aussi un contrôle de validité qui consiste à rechercher un élément obligatoire (comme un verbe à l'infinitif, etc.) à l'intérieur de ce que le système estime être une proposition. A nouveau, si le contrôle de validité échoue le *RESCUE operator* fait en sorte que les traitements effectués par *SEGMENT* soient réitérés avec cependant moins de restrictions sur les règles à appliquer.

2.2.5. NOMA - *Analyse des groupes nominaux*

NOMA travaille sur les segments qui lui sont fournis par le module précédent et essaie d'y identifier des groupes nominaux selon le principe du plus long *matching*. La routine principale de *NOMA* s'occupe des groupes nominaux simples de format PREP+ART+NOM, ART+GROUPE ADJ, etc. D'autres routines sont destinées à des traitements plus spécifiques comme les appositions, les numéraux, etc. se basant sur la présence de signes de ponctuation, de mots déterminés comme *par exemple* ou sur les valences.
Des tests de compatibilité entre groupes identifiés dans une même phrase sont ensuite effectués et il est possible de tester de nouvelles combinaisons (par ordre de plausibilité) si les anciennes sont incompatibles. Dans le cas où le système ne peut identifier une structure globale, le *Rescue operator* est à nouveau invoqué et l'analyse recommence.

2.2.6. VERA - *Analyse de groupes verbaux*

De la même façon que *NOMA* traite les groupes nominaux, *VERA* traite les groupes verbaux. *VERA* utilise les résultats de *NOMA* et est également séparé en différentes routines spécialisées par type de groupe verbal (verbes composés, verbes avec auxiliaire, etc.).

2.2.7. KOMA - *Combinaisons de groupes nominaux et des groupes verbaux*

KOMA a pour but d'identifier la structure de valence liée à chaque proposition. Pratiquement, il s'agit de déterminer quels groupes nominaux (identifiés par *NOMA*) jouent le rôle d'arguments de verbe des groupes verbaux identifiés par *VERA*. Cela se fait en testant les cas syntaxiques des groupes nominaux, leur nombre ainsi que certaines restrictions sémantiques. *KOMA* est en fait le module qui produit une analyse structurelle de la phrase.

2.2.8. SEDAM - *"Désambiguätion" sémantique*

SEDAM dispose d'un « dictionnaire sémantique » dans lequel on a attaché à chaque entrée (les noms et quelques pronoms uniquement) des caractéristiques sémantiques de deux types :

- universelles (humain, abstrait, animé, etc.) et organisées de manière hiérarchique,
- relatives à une langue donnée et se rapportant à des groupes nominaux (lieux géographiques, profession, etc.).

A chaque entrée est également associé un ensemble de règles qui peuvent produire des changements, des effacements et des insertions dans la structure. *SEDAM* a en fait pour but de donner une interprétation à des structures syntaxiques qui sont sémantiquement ambiguës. Les principales utilités sont :

- le traitement des noms se rapportant à des *process nouns* qui peuvent être agents (par exemple human understanding), objets (par exemple, satellite launching), etc. selon le cas,
- le traitement des homographes de même catégorie lexicale.

Ces traitements sont manifestement motivés par la diminution des problèmes de transfert.

*2.3. Le transfert (*TRANSFERT*)*

A la base, le transfert se fait uniquement au niveau lexical en établissant des correspondances « mot-à-mot » entre mots de la langue source et mots de la langue cible. Il existe cependant des procédures spécialisées pour traduire la négation, les relations (cas de surface) dans les groupes nominaux.

2.4. La synthèse

2.4.1. SEMSYN - Synthèse au niveau sémantique

SEMSYN a principalement pour but de générer les expressions idiomatiques, traduire les prépositions en fonction du nom ou du verbe qui les gouverne.

2.4.2. SYNSYN - Synthèse au niveau syntaxique

Un des buts de SYNSYN est d'établir l'ordre des mots dans la structure de surface, ainsi que les terminaisons des cas de surface. Certaines procédures sont chargées d'effectuer des améliorations stylistiques surtout au niveau des groupes nominaux pour que, par exemple, *translation system* soit traduit par *système de traduction*. Parmi les autres rôles de SYNSYN on trouve la génération des pronoms et des adjectifs possessifs par accord avec leurs antécédents.

2.4.3. MORSYN - Synthèse au niveau morphologique

Ici, les éléments exprimés en termes de *racine* + *caractéristiques morphologiques* sont transformés en chaînes de caractères de la langue cible. Enfin MORSYN s'occupe également des élisions, de l'utilisation des majuscules et de la ponctuation.

3. CRITIQUE DU SYSTEME

Susy est un système de transfert assez primitif. Il ne dispose pas d'un formalisme de haut-niveau dans lequel il est possible d'écrire des règles dont l'exécution est prise en charge pas un module existant. Ceci se retrouve dans l'architecture de Susy qui montre un grand nombre d'étapes successives divisées plus ou moins séquentiellement en traitements linguistiques "simples" et même en procédures que l'on peut qualifier d'*ad hoc*. Cette subdivision n'est pas une séparation par niveaux linguistiques comme, par exemple, dans Eurotra, mais plutôt par tâche spécifique à exécuter par le système. Susy souffre donc au départ d'un manque de

généralité : on ne peut traiter un phénomène linguistique que si une procédure (FORTRAN) a été écrite dans cette optique.

Cependant, l'architecture de Susy, munie du mécanisme de *fail-soft RESCUE* (voir section 2.1 de ce chapitre) présente un grand avantage : elle permet de traiter des phénomènes linguistiques de manière non stratifiée linguistiquement (contrairement à ce qu'on trouvait dans Eurotra). Malheureusement, cela se fait au prix de retours en arrière qui introduisent une certaine inefficacité du système.

Le niveau des connaissances linguistiques, quant à lui, ne dépasse pas le niveau sémantique des "case frames" et on retrouve donc le problème de manque de "compréhension" du texte traduit.

L'approche linguistique montre néanmoins une certaine cohérence. On remarquera notamment le fait que l'on travaille toujours sur une représentation homogène de la phrase tout au long de l'analyse (arbre de dépendance), ce qui n'est pas le cas chez Logos d'ailleurs. Nous reprochons cependant le recours trop fréquent à des règles de type *ad hoc* (recherche de tel mot particulier pour repérer tel type de proposition etc.) ainsi que l'emploi de considérations statistiques là où l'utilisation de règles de plus haut niveau eut été plus judicieuse.

Susy se veut être un système multilingue mais dont l'étendue est beaucoup plus limitée qu'Eurotra (allemand, russe, français, anglais) et l'approche *transfert* peut donc se justifier.

Au niveau des données lexicales, signalons que, contrairement aux données grammaticales, celles-ci sont clairement séparées de la partie algorithmique du système. Deux lexiques monolingues servent respectivement à l'analyse et à la génération et on trouve tout naturellement un lexique de transfert bilingue. A côté de cela, il faut constater la présence d'un "dictionnaire sémantique" bilingue utilisé tant dans l'analyse que la génération (voir figure 6.1 ci-dessus). Il en résulte que l'analyse et la génération ne sont pas entièrement monolingues et, dans un véritable système de *transfert*, c'est le module de transfert qui devrait avoir accès à ce dictionnaire.

Enfin, au niveau de l'utilisation pratique de Susy, deux faits sont à souligner : premièrement, l'existence d'une pré-édition facultative (voir chapitre II, section 3.7.1) et, ensuite, le fait que, bien que la post-édition ne soit pas automatisée, elle est facilitée par les contrôles de consistance pratiqués sur les entrées/sorties des modules et qui peuvent aider à identifier les problèmes restants.

CHAPITRE VII : Systran

Après une brève présentation du système (section 1), nous allons décrire les éléments qui constituent le processus de traduction (section 2). Nous conclurons par une critique de Systran (section 3) et nous en profiterons pour faire une comparaison avec Logos. Les documents qui ont servis de base pour la description de ce système sont [PIGGOT] et le chapitre XX de [HUTCHINS].

1. PRESENTATION DU SYSTEME

Le premier système Systran traduisant du russe vers l'anglais fut développé dès 1968 et est utilisé par l'US Air Force depuis 1970. Son concepteur, Peter Toma, travailla sur SERNA (implémentation du système GAT) à l'université de Georgetown avant de créer sa propre compagnie en 1962 et de développer deux systèmes concernant des domaines particuliers : AUTOTRAN (énergie atomique) et TECHNOTRAN (médecine).

En 1975, le prototype du système Systran anglais-français fut présenté aux communautés européennes. Jugeant le résultat satisfaisant, un contrat fut établi afin de développer des systèmes traduisant les différentes langues européennes. Le tableau 7.1 fournit la liste des paires de langues disponibles et en développement en 1992.

Disponibles	En développement
Anglais ⇔ Français	Anglais ⇔ Chinois
Anglais ⇔ Allemand	Anglais ⇔ Coréen
Anglais ⇔ Japonais	Anglais ⇒ Arabe
Anglais ⇔ Russe	Anglais ⇒ Danois
Anglais ⇔ Espagnol	Anglais ⇒ Néerlandais
Anglais ⇒ Italien	Anglais ⇒ Finnois
Anglais ⇒ Portugais	Anglais ⇒ Norvégien
Allemand ⇒ Français	Anglais ⇒ Suédois
Allemand ⇒ Italien	Français ⇒ Néerlandais
Allemand ⇒ Espagnol	Français ⇒ Allemand
	Français ⇒ Italien
	Espagnol ⇒ Anglais
	Portugais ⇒ Anglais

Tableau 7.1. : Paires de langues disponibles et en développement pour Systran

Systran ne définit aucune contrainte au niveau de la structure grammaticale ou à propos de la terminologie à utiliser. Il n'a recours à l'intervention humaine qu'au moment de la post-édition. Le but des concepteurs est donc de disposer d'un système *general purpose* qui minimise le travail de post-édition.

2. *LE PROCESSUS DE TRADUCTION*

Le nombre élevé de paires de langues nécessite une modularité importante de l'ensemble du processus de traduction. Celui-ci a été divisé en quatre grandes étapes (pré-traitement, analyse, transfert, synthèse) afin d'identifier une étape qui soit invariante pour une même langue source (analyse) et une qui soit invariante pour une même langue cible (synthèse). Dans l'architecture présentée à la figure 7.1, on peut constater que ces étapes sont elles-mêmes composées de modules ayant des tâches spécifiques. Ces tâches peuvent consister à accéder à un dictionnaire et/ou à faire appel à des algorithmes linguistiques.

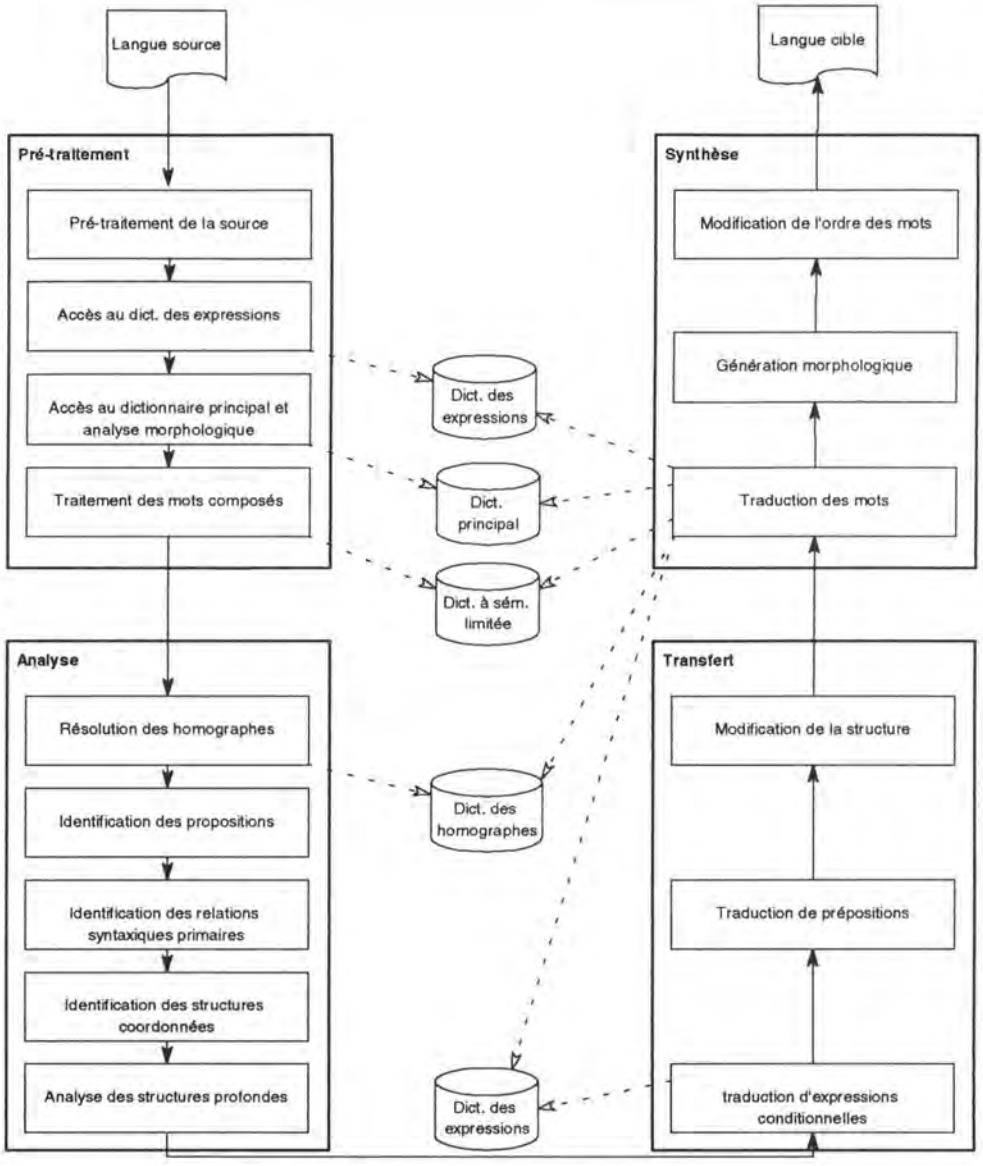


Figure 7.1. : L'architecture du système Systran

2.1. Les dictionnaires

Systran dispose d'un dictionnaire principal et de plusieurs dictionnaires contextuels qui sont utilisés à différents stades de la traduction. Les entrées du dictionnaire principal sont constituées des racines des mots qui existent dans la langue source (sauf dans le cas de l'anglais où il existe une entrée pour chaque forme fléchie en raison de leur nombre relativement peu élevé). Il existe une entrée différente pour tous les homographes grammaticaux distincts. Ce dictionnaire fournit pour chaque entrée une description morphologique (genre, nombre, ...), syntaxique (transitivité, accord, ...) ainsi que sémantique par l'intermédiaire de types ("animé", "abstrait", ...) et de marqueurs ("produit", "nourriture", ...). Il contient également la traduction

par défaut de chaque entrée accompagnée par des informations grammaticales nécessaires pour une génération correcte.

Il existe 4 autres dictionnaires dont les entrées peuvent être constituées de plusieurs mots et qui ont pour but d'adapter la traduction au contexte :

- *le dictionnaire des expressions* : il traite les expressions invariantes dont la traduction est constante (exemples: *on the hand, in order to*, ...).

- *le dictionnaire à sémantique limitée* : il définit la portée des relations syntaxiques à l'intérieur des groupes nominaux afin de guider la traduction (exemples: *machine translation, pomme de terre*, ...)

- *le dictionnaire des homographes* : il donne la liste des informations contextuelles syntaxiques nécessaires à la résolution de certaines ambiguïtés dues aux homographes (exemple: la forme transitive du verbe *"prendre"* est toujours suivie d'un déterminant sauf dans certains cas particuliers tels que *"prendre note"*).

- *le dictionnaire à sémantique conditionnelle* : il détermine de manière définitive la traduction d'un mot selon des critères sémantiques et syntaxiques.

Les autres dictionnaires présents dans Systran sont les dictionnaires analytiques. Ils contiennent les exceptions aux règles syntaxiques générales qui s'appliquent à certains mots. Ils peuvent être accédés à divers moments de la traduction.

2.2. L'étape de pré-traitement

Cette étape s'applique à l'ensemble du texte, elle consiste à analyser le format, à identifier les éléments par l'accès à plusieurs dictionnaires et à effectuer une analyse morphologique. Ces tâches sont remplies par quatre modules successifs :

- *pré-traitement de la source* : charge le texte source et extrait les informations concernant le format.

- *accès au dictionnaire des expressions* : attribue une catégorie grammaticale unique aux expressions identifiées.

- *accès au dictionnaire principal et analyse morphologique* : recherche tous les mots qui n'ont pas reçu de catégorie lexicale lors du module précédent. Cette recherche nécessite une analyse morphologique sauf dans le cas de l'anglais puisque toutes les formes fléchies ont une entrée spécifique dans le dictionnaire. L'analyse morphologique permet également d'obtenir des informations grammaticales sur les mots absents du dictionnaire.

- *Traitement des mots composés* : identifie les mots composés en accédant au dictionnaire à sémantique limitée. Cette manière de traiter les séquences de mot peut entraîner des erreurs (par exemple, il n'est pas possible de faire traduire la phrase: *"il parla à la femme de ménage"*, par: *"He spoke to the woman about housekeeping"*).

Les trois étapes suivantes (analyse, transfert et synthèse) effectue leurs tâches phrase par phrase.

2.3. L'étape d'analyse

Cette étape est pratiquement invariante pour une même langue source. Elle détermine les catégories grammaticales et identifie les relations syntaxiques qui existent entre les différents éléments. Elle s'effectue par la succession de cinq modules:

- *résolution des homographes* : analyse les catégories grammaticales possibles en utilisant le dictionnaire des homographes. La stratégie consiste à effectuer un seule passe en considérant les mots adjacents. Dans le cas où les règles dont dispose le système ne parviennent pas à déterminer la *part-of-speech* correcte, c'est la plus courante qui est sélectionnée.

- *identification des propositions* : insère des marqueurs indiquant les débuts et fins de propositions identifiées par la présence de ponctuations, de conjonctions, de pronoms relatifs, ...

- *identification des relations syntaxiques primaires* : définit les relations entre un nom et ses modificateurs, un verbe et son objet, ... Ce module a plusieurs autres tâches telles que rechercher le verbe principal, repérer les superlatifs, ...

- *identification des structures coordonnées* : détermine les mots qui font partie d'une énumération sur base d'informations syntaxiques et sémantiques.

- *analyse des structures profondes*.

Toutes les informations recueillies sont placées en mémoire centrale où un certain nombre de bytes sont réservés pour chaque mot du texte. Tous les bytes appartenant à un même élément sont numérotés, ils correspondent chacun à un type d'information défini par convention selon leur numéro. Les algorithmes font directement référence à ces bytes par un système de pointeur.

2.4. L'étape de transfert

L'étape de transfert est effectuée par trois modules :

- *traduction d'expressions conditionnelles* : modifie la traduction de certains mots selon les caractéristiques syntaxiques identifiées lors de l'étape d'analyse.

- *traduction des prépositions* : traite les prépositions qui n'ont pas été prises en charge par les modules précédents.

- *modification de la structure* : utilise des routines lexicales afin de modifier la structure de la traduction des phrases contenant certaines catégories syntaxiques ou sémantiques des mots.

2.5. L'étape de synthèse

L'étape de synthèse comporte également trois modules :

- *traduction des mots* : assigne la traduction par défaut du dictionnaire aux mots qui n'ont pas été traités lors des modules précédents.

- *génération morphologique* : fléchit les mots selon les informations structurelles concernant le genre, le nombre, ...

- *modification de l'ordre des mots* : détermine l'ordre dans lequel les mots doivent être agencés. Ce module peut également effectuer d'autres tâches qui dépendent de la langue cible (les élisions pour le français par exemple).

Dans cette dernière étape, les tâches à effectuer font souvent référence à la source, seul le module de génération morphologique est indépendant du texte de départ.

3. *CRITIQUE DU SYSTEME*

Les premiers prototypes de Systran étaient basés sur une approche directe. La nécessité d'améliorer la qualité des traductions et d'évoluer vers un système multilingue a forcé les responsables du projet à se diriger vers une approche de type transfert. Actuellement, même si beaucoup de progrès ont été fait dans ce sens, il reste encore plusieurs objections qui peuvent être émises quant à la réalisation de cet objectif :

- Il n'existe aucune étape invariable pour la langue cible dans le système, cela signifie que l'étape de génération dépend toujours de la langue source concernée. L'introduction d'une nouvelle paire de langues nécessite donc l'écriture d'un module de génération spécifique, ce qui constitue un frein important au développement d'un système multilingue.

- Les informations lexicales sont placées dans des dictionnaires auxquels accèdent les modules d'analyse ainsi que les modules de génération. L'utilisation d'un dictionnaire bilingue est un des vestiges de la méthode directe qui était employée auparavant.

- Il n'y a pas d'analyse complète de la phrase. Seules les relations entre les éléments de la phrase sont identifiées, les groupes verbaux et les groupes nominaux ne sont pas reconnus. La résolution des homographes est entièrement effectuée sans avoir la moindre donnée sur la présence de propositions (cette tâche étant remplie par le module suivant). De plus, il n'existe aucune représentation disponible permettant de visualiser le résultat de l'analyse. Cependant, il faut remarquer que le système fait référence aux théories linguistes lorsqu'il distingue l'analyse des structures de surface et des structures profondes.

- La prise en considération de suites de mots comme entrées dans le dictionnaire entraîne une certaine rigidité du système car celui-ci n'analysera pas en profondeur cette partie de la phrase.

- Les marqueurs sémantiques sont attribués aux éléments de manière empirique et ils n'existe aucune hiérarchie (telle que les *SAL codes* de Logos) entre ces codes. Il en résulte une grande incohérence dans la base de données.

Du point de vue de l'écriture des algorithmes, malgré la création (assez récente) d'un macro-langage, la tâche à réaliser se situe toujours à un niveau très bas.

Systran et Logos ont suivi la même méthode empirique de développement des paires de langues. Par conséquent, ils sont tous deux remarquables par leur complexité. Cependant, Logos diffère fortement de Systran à deux points de vue. Premièrement, lors de l'étape d'analyse, le système Logos prend en compte les informations concernant l'identification de propositions et il peut même effectuer des recherches en avant. Il en résulte un taux d'erreur plus faible dans la résolution des homographes, ce qui diminue déjà de manière significative le nombre d'erreurs de traduction. Deuxièmement, Logos dispose d'une taxonomie pour ses codes sémantiques. Ces codes ont été définis dès le début des recherches, cela signifie qu'ils n'ont pas été attribués au cours du développement selon les besoins occasionnels.

Ces différences font que le système Logos n'a pas encore atteint le point où toute modification améliorant certaines traductions interfère avec des traductions qui étaient correctes auparavant. Au contraire, il semble que Systran ait maintenant beaucoup de mal à faire progresser la qualité de ses traductions de manière significative.

CHAPITRE VIII : TAUM-Meteo

Dans ce chapitre, nous allons décrire l'ensemble du système de traduction (section 2) après avoir décrit la naissance du système TAUM-Meteo (section 1) et nous conclurons par la critique de ce système (section 3). Pour cette présentation, nous nous sommes basés sur [ISABELLE] ainsi que sur le chapitre XXII de [HUTCHINS].

1. PRESENTATION DU SYSTEME

C'est à l'université de Montréal, sous la direction de Guy Rondeau, que le premier prototype du système TAUM (Traduction Automatique de l'Université de Montréal) fut développé entre 1968 et 1971. Ce système traduit de l'anglais vers le français et est basé sur une approche transfert.

En 1975, TAUM s'engage à fournir un système de traduction de prévisions météorologiques. Il s'agit d'une tâche idéale pour la Traduction Automatique du point de vue pratique (le travail est très répétitif et peu gratifiant pour les linguistes), mais surtout du point de vue linguistique. En effet, les phrases ne concernent qu'un domaine particulier, les bulletins respectent tous un format pré-déterminé (voir exemple ci-dessous) et le style utilisé est proche du style télégraphique. Les phrases à traduire appartiennent donc à un sous-langage, ce qui explique une simplification très importante des étapes qui forment l'approche transfert. Cette restriction du domaine d'application permet d'être beaucoup plus optimiste quant à la qualité de la traduction finale.

La première version de TAUM-Meteo, utilisée à partir de 1977, a été remplacée en 1984 par une seconde version plus rapide et plus sûre. En 1989, un système identique traduisant du français vers l'anglais fut également mis à la disposition du département de l'environnement.

Exemple de bulletins météorologiques

FPCN11 CYYZ 311630

FORECASTS FOR ONTARIO ISSUED BY ENVIRONMENT CANADA AT 11.30 AM EST WEDNESDAY MARCH 31ST 1976

FOR TODAY AND THURSDAY.

METRO TORONTO

WINDSOR.

CLOUDY WITH A CHANCE OF SHOWERS TODAY AND THURSDAY.

LOW TONIGHT 4. HIGH THURSDAY 10.

OUTLOOK FOR FRIDAY... SUNNY

END

2. *LE PROCESSUS DE TRADUCTION*

Les recherches concernant le projet TAUM étaient axées sur une approche de type transfert. Cependant, la restriction du domaine d'application dans le cadre du projet TAUM-Meteo a entraîné le retour à une approche plus directe. La phase de transfert a été purement et simplement supprimée, comme on peut le voir sur la figure 8.1. Le sous-langage concerné étant tellement limité, il était plus facile d'effectuer les tâches théoriquement dévolues à l'étape transfert lors des étapes de synthèse, et surtout, d'analyse.

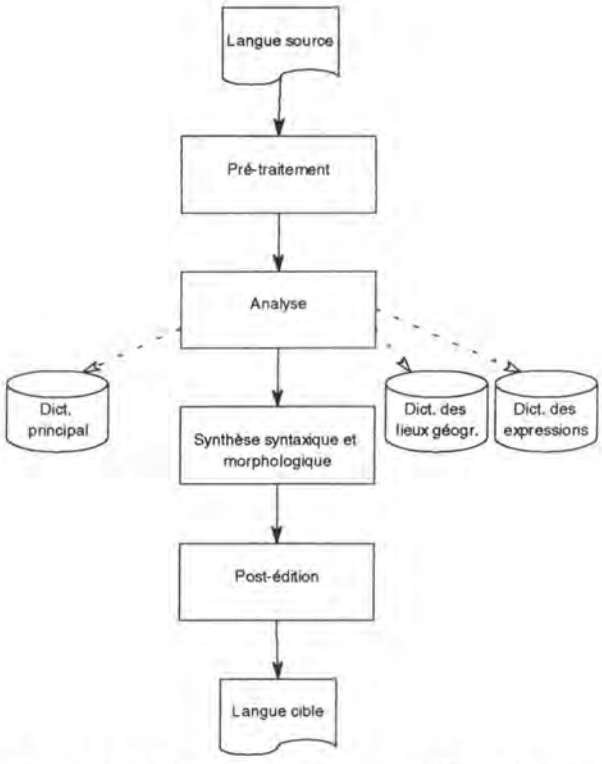


Figure 8.1. : L'architecture du système TAUM-Meteo

2.1. Les dictionnaires

Le système TAUM-Meteo dispose de trois dictionnaires:

- *dictionnaire des expressions* : contient des séquences de mots qui doivent être traitées comme un élément unique.

- *dictionnaire des lieux géographiques* : contient les noms de tous les lieux géographiques qui disposent d'une traduction spécifique.

- *dictionnaire principal* : contient, pour chaque entrée, des informations relatives à la catégorie lexicale, la sémantique, la traduction dans la langue cible et des informations morphologiques sur cette traduction. Il dispose également d'informations sur les caractéristiques propres à certaines catégories lexicales (transitivité des verbes, ...). Il existe une entrée pour chaque forme fléchie et pour toutes les traductions possibles dans la langue cible.

2.2. L'étape de pré-traitement

L'étape de pré-traitement est rendue nécessaire par le besoin du système TAUM-Meteo de recevoir les phrases à traduire selon une syntaxe très particulière. Son rôle principal consiste à insérer des marques délimitant les mots et les phrases.

Exemple de bulletins météorologique pré-traité

-01- $(TORONTO,2) + FORECASTS + FOR + ONTARIO + ISSUED + BY + ENVIRONMENT + CANADA + AT + 11 + H + 30 + AM + EST + WEDNESDAY + MARCH + 31ST + 1976 + FOR TODAY + AND + THURSDAY+ . -02-/

-01- $(TORONTO,3) + METRO + TORONTO + , + WINDSOR + . -02-/

-01- $(TORONTO,4) + CLOUDY + WITH + A + CHANCE + OF + SHOWERS + TODAY + AND + THURSDAY+ . -02-/

-01- $(TORONTO,5) + LOW + TONIGHT + 4 + . -02-/

-01- $(TORONTO,6) + HIGH + THURSDAY + 10 + . -02-/

-01- $(TORONTO,7) + OUTLOOK + FOR + FRIDAY+ = + SUNNY + . -02-/

2.3. L'étape d'analyse

L'étude du sous-langage utilisé pour écrire les bulletins météorologiques a mené à l'identification de cinq types de structure de phrase. Toutes les phrases analysées peuvent être reconnues comme appartenant à l'un des 5 types suivant :

MET0 : correspond à une liste de lieux géographiques.

MET1 : `( C ({Adj/GN},[Cmod]) , [T] , [1] )`
représente les phrases qui expriment des conditions météorologiques.

MET2 : `( GN ({High/low/...}) , [T] , [1] , GN ({TEMP}) , [T] , [L] )`
représente les phrases qui expriment les températures maximum et minimum .

MET3 : `( GN (outlook for T), C ({Adj/GN},[Cmod]) , [T] , [L] )`
représente les phrases qui expriment les prévisions météorologiques.

MET4 : correspond à la phrase d'en-tête indiquant la provenance du bulletin.

Le processus utilisé par le système TAUM-Meteo pour l'analyse de la langue source est équivalent à celui d'un système dit "de production". Il s'agit d'un système qui travaille sur une représentation sous forme de *chart* des données et qui applique une suite de règles dont l'ordre n'est pas pré-défini. La première phase consiste à accéder aux différents dictionnaires et à créer le *chart* correspondant à la phrase analysée.

L'analyse est réalisée en appliquant une suite de règles qui sont composées de deux éléments correspondant aux conditions de leur application et aux actions à effectuer. Les conditions sont exprimées sous forme de *charts* qui représentent les éléments qui doivent se trouver dans la structure analysée (les conditions permettent de tenir compte de la sémantique). Les actions n'altèrent pas le *chart* initial, cela signifie que les règles qui n'ont pas été sélectionnées dans un premier temps pourront toujours être appliquées plus tard. Il est fréquent que la structure du *chart* corresponde à plusieurs règles. Dans ce cas, le système en sélectionne une et effectue un retour en arrière si elle ne s'avère pas être la bonne (recherche en profondeur d'abord).

Les règles sont appliquées jusqu'à l'obtention d'une structure correspondant à l'une des cinq formes de phrase identifiées ci-dessus.

Du point de vue linguistique, l'analyse est réalisée en trois phases. La première doit effectuer diverses tâches:

- identifier les dates, les heures et les degrés.
- transformer les expressions telles que *"in the low 20s"* en *"21 à 25"*.
- transformer la notation am/pm en notation 24h.
- identifier et traduire les groupes nominaux et les adverbes qui expriment les périodes de la journée ou les lieux géographiques.

La seconde phase identifie et traduit les groupes nominaux qui restent et qui correspondent à des conditions météorologiques. Les traductions tiennent compte de caractéristiques sémantiques afin d'être les plus adéquates possible. Enfin, la dernière phase détecte les relations grammaticales entre les différents groupes afin d'identifier les structures pré-définies.

2.4. L'étape de synthèse

Cette étape est réalisée de la même manière que la précédente, c'est-à-dire que l'application d'ensembles de règles ayant des rôles différents tels que:

- déplacement des mots dans la phrase afin de respecter l'ordre requis par la langue cible.
- sélection de la traduction correcte des propositions.
- génération morphologique.
- réalisation des élisions.

2.5. L'étape de post-édition

Certaines erreurs, dues en grande partie aux bruits sur les communications ou à des fautes de frappe, peuvent apparaître dans la traduction. Ces phrases incorrectes sont facilement repérées par le système en raison du vocabulaire très restreint utilisé dans les bulletins météorologiques. Dès que le système rencontre un mot inconnu, il en déduit qu'il y avait une erreur dans les données reçues. Les phrases contenant des mots absents du dictionnaire sont donc automatiquement envoyées à des linguistes.

3. CRITIQUE DU SYSTEME

Comme nous l'avons déjà expliqué, bien que le système TAUM était basé sur une approche transfert, l'application au domaine météorologique a eu pour conséquence la suppression du module de transfert. il ne faut cependant pas croire pour autant qu'il s'est transformé en un système de type *interligua*. Il n'existe aucune représentation intermédiaire, cela n'aurait d'ailleurs aucun sens puisque le système TAUM-Meteo n'a jamais eu la prétention de devenir multilingue.

- Le cas du système TAUM-Meteo est particulièrement intéressant dans le sens où il reste actuellement le seul système fournissant une traduction de très grande qualité. Malheureusement, il n'offre pas de grandes perspectives pour d'autres langues ou dans d'autres domaines dont le champ d'application serait un peu plus vaste, et ce pour plusieurs raisons :

- Le but premier étant de traduire uniquement de l'anglais vers le français, le module d'analyse à été écrit en conséquence.

- Le système ne doit faire face à des homographes que dans de rares occasions.

- Le nombre peu élevé de règles permet d'effectuer des recherches en profondeur d'abord et de revenir en arrière en cas d'erreur. Une telle méthode serait beaucoup plus difficile à gérer et à implémenter si le nombre de règles devait fortement augmenter, ce qui serait le cas si le domaine d'application était plus vaste.

- Un élargissement du champ d'application serait suivie par une complexification du sous-langage concerné.

Cette affirmation a été vérifiée lorsque les responsables du projet TAUM ont tenté d'appliquer la même méthode au projet se rapportant aux manuels de maintenance de l'aviation. Dans ce domaine aussi il était possible de déterminer un sous-langage, mais il était déjà beaucoup plus complexe que celui des prévisions météorologiques. Cela a rapidement mené à l'abandon de ce projet.

CHAPITRE IX : Metal

Après une brève présentation du système (section 1), nous allons décrire les éléments qui constituent le processus de traduction (section 2). Nous conclurons par une critique de Metal (section 3) et nous en profiterons pour faire une comparaison avec Logos. Le chapitre XXV de [HUTCHINS] a servi de base pour la description présentée ci-dessous.

1. PRESENTATION DU SYSTEME

Metal fut développé dès 1961 à l'université du Texas sous la direction de Winfried Lehmann. Jusque 1968, les travaux consistaient à étudier les syntaxes de l'anglais et de l'allemand afin de construire un traducteur allemand-anglais disposant de règles de traduction syntaxiques réversibles. C'est en 1978 que Siemens commença à supporter financièrement le projet Metal. Cette date marqua également l'abandon de l'approche *interlingua* pour passer à une approche de type transfert.

Plusieurs autres paires de langues ont été développées depuis, la liste en est présentée dans le tableau 9.1.

Paires de langue disponibles		
Allemand	⇔	Anglais
Français	⇔	Néerlandais
Allemand	⇒	Français
Allemand	⇒	Danois
Allemand	⇒	Espagnol

Tableau 9.1. : Paires de langues disponibles pour Metal

2. LE PROCESSUS DE TRADUCTION

L'évolution de Metal vers un système multilingue a entraîné une forte modularité à plusieurs niveaux. Le premier niveau se situe entre algorithme et donnée linguistiques (dictionnaires et règles) puisque l'introduction d'une nouvelle paire de langues ne nécessite aucune modification des algorithmes qui gèrent les accès aux différents dictionnaires et l'application des règles linguistiques. Le second niveau met en évidence la présence de trois étapes distinctes (voir figure 9.1) dans le processus de traduction (il s'agit des étapes classiques d'analyse, de transfert et de génération).

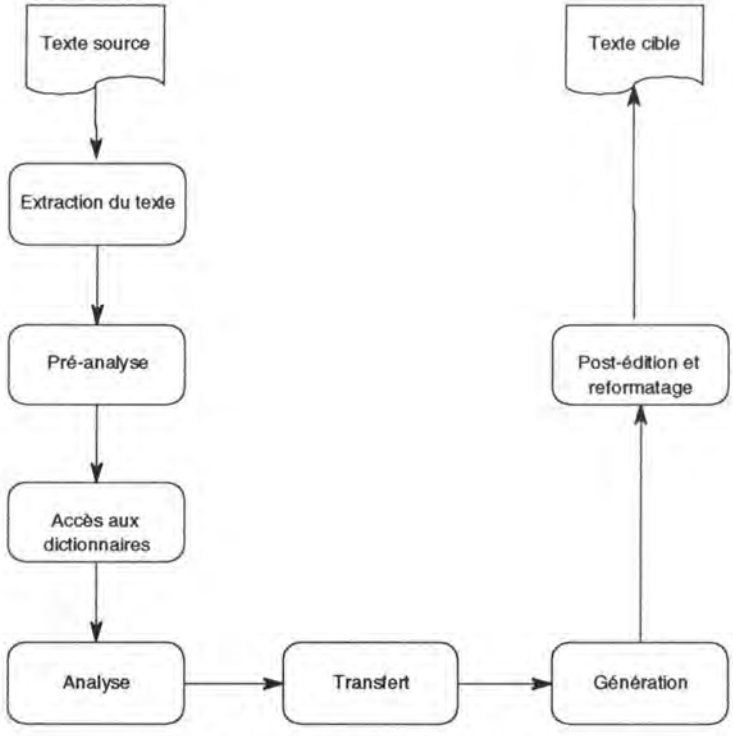


Figure 9.1. : Architecture du système Metal

2.1. Les dictionnaires

Metal dispose de dictionnaires monolingues (un pour la langue source et un pour la langue cible) et d'un dictionnaire bilingue.

Les dictionnaires monolingues fournissent les informations de base sur la morphologie, la syntaxe et la sémantique. L'utilisateur peut utiliser la hiérarchie qui existe parmi l'ensemble du vocabulaire afin de sélectionner la traduction qui correspond à ses besoins (de cette manière, le mot *"lastwagen"* sera traduit par *"truck"* aux Etats-Unis et par *"lorry"* en Grande Bretagne).

Il existe une entrée pour toutes les formes fléchies. Les informations propres à chaque entrées sont représentées par une suite de codes formés de caractères.

C'est le dictionnaire bilingue qui fait le lien entre les entrées du dictionnaire de la langue source et de celui de la langue cible.

Exemple d'entrées de dictionnaires

dictionnaire allemand :

(ausgabe	cat	(nst)
	alo	(ausgabe)
	plc	(wi)
	tag	(dp)
	cl	(p-n s-0)
	gd	(f)
	sx	(n)
	ty	(abs dur)

dictionnaire anglais :

(output	cat	(nst)
	alo	(ouput)
	plc	(wi)
	tag	(dp)
	cl	(p-s s-01)
	on	(vc)
	sx	(n)

dictionnaire bilingue :

(Ausgabe (nst dp) 0 output (nst dp) 0)

Dans le cas où plusieurs traductions sont possibles suivant le contexte sémantico-syntaxique, les différentes alternatives sont examinées dans un ordre prédéfini par des valeurs qui se trouvent dans le dictionnaire (ces valeurs sont à 0 dans l'exemple ci-dessus puisqu'il n'existe aucune alternative).

2.2. Les règles

Les règles du système Metal sont écrites en Lisp, elles comportent les tâches relatives à la morphologie ainsi qu'à la syntaxe et elles combinent les actions à effectuer lors de l'analyse et lors du transfert. Ces règles sont composées de quatre éléments :

- un double test dont le rôle est de définir si la règle doit être appliquée. Le premier test spécifie les conditions morphologiques et syntaxiques, tandis que le second vérifie qu'il existe une certaine cohérence parmi les caractéristiques des éléments concernés. Ce dernier test est réalisé par la comparaison des marques syntaxiques qui se trouvent dans le dictionnaire.

- la spécification de la structure qui doit être construite afin de modéliser par un arbre l'analyse effectuée par le système. Cette partie de la règle défini également les informations qui doivent être attachées à chaque nouveau noeud de l'arbre et, éventuellement, modifier les informations présentes dans les autres noeuds.

- la spécification des transformations qui doivent être appliquées à l'arbre.

- la spécification des opérations à effectuer lors de l'étape de transfert. En effet, une fois que l'étape d'analyse est terminée, l'étape de transfert se résume à effectuer, de manière *top-down*, toutes les opérations spécifiées par les règles qui ont été appliquées lors de la phase d'analyse.

2.3. Extraction du texte

Les documents à traduire peuvent contenir des tableaux, des graphiques ou d'autres composants qui ne concernent pas la traduction. Ce premier module a donc pour rôle de transmettre les phrases au système de traduction et de mémoriser le format afin de pouvoir reconstruire le document selon le même format avec les phrases traduites.

2.4. Pré-analyse

L'étape de pré-analyse accède à la base de données des dictionnaires afin de créer trois listes contenant les mots simples inconnus, les mots composés inconnus (utilisées principalement pour l'allemand) et les mots techniques présents dans les dictionnaires mais dont l'adéquation au texte doit être vérifiée.

2.5. Accès aux dictionnaires

Ce module n'a pas d'autres rôles que d'extraire des dictionnaires les informations concernant les éléments de la phrase.

2.6. L'étape d'analyse

La première tâche de l'étape d'analyse consiste à faire une étude morphologique des éléments de la phrase afin d'identifier les racines et leurs affixes.

La seconde consiste à établir un classement parmi les différentes analyses syntaxiques et grammaticales possibles de la phrase. Ce classement est réalisé sur base de poids attribués aux différents arbres élaborés. Ces poids sont calculés d'après les valeurs de préférence qui sont attachées à toutes les entrées lexicales ainsi qu'à toutes les structures grammaticales. Plusieurs arbres sont construits par l'application de règles selon une logique *bottom-up* (ce sont les règles concernant les niveaux inférieurs qui sont exécutées en premier). Tous les arbres possibles ne sont pas pris en compte, ceux qui sont considérés comme improbable en raison de leur poids particulièrement faible sont éliminés automatiquement.

A la fin de ce processus, il ne reste plus qu'à sélectionner l'arbre dont le poids est le plus élevé pour déterminer la structure de l'analyse réalisée par le système. Il faut noter une particularité importante des règles qui sont appliquées : elles ne disposent d'aucune condition relative à la sémantique.

2.7. L'étape de transfert

Comme nous l'avons vu, toutes les règles appliquées lors de l'étape précédente disposent d'un ensemble de fonctions qui indiquent les opérations à effectuer lors de l'étape de transfert. Ces fonctions étant attachées au noeuds concernés, il suffit de les appliquer, de manière *top-down*, pour que l'étape de transfert soit réalisée.

Ces fonctions effectuent des opérations complexes qui font interagir le dictionnaire bilingue et la structure identifiée lors de l'étape d'analyse. Le choix des traductions les plus adaptées se fait sur base de critères syntaxiques et sémantiques.

2.8. L'étape de génération

Dans un premier temps, l'étape de génération avait pour seul but de générer de manière morphologiquement correcte la phrase cible. Son rôle a été étendu dans le but de faire évoluer Metal vers un système de type multilingue.

2.9. Post-édition et reformatage

La post-édition peut être effectuée simplement sur le texte brut ou mis en forme selon les critères retenus du document de départ. Cette correction des erreurs de traduction peut être réalisée phrase par phrase ou à un niveau plus élevé si le linguiste le désire.

3. CRITIQUE DU SYSTEME

Dès l'arrivée de Siemens en 1978, l'approche utilisée pour le développement du système Metal fut de type transfert. En effet, le premier système traduisant de l'allemand vers l'anglais disposait déjà de plusieurs caractéristiques qui ne laissent pas de doutes quant à l'approche choisie :

- La division entre données et algorithme est nette : la modification des règles et des dictionnaires suffit à ajouter une paire de langues, aucune modification ne doit être effectuée au niveau des procédures informatiques.

- La séparation au sein des dictionnaires est très claire : il existe un dictionnaire monolingue pour chaque langue et un dictionnaire bilingue (utilisé uniquement lors de la phase de transfert) pour chaque paire de langues disponible.

- L'étape d'analyse est entièrement réalisée avant que ne débute l'étape de transfert. De plus, il existe à ce moment une représentation détaillée de l'analyse effectuée.

Cependant, il restait toujours une objection quant à la pureté de l'approche transfert : l'étape de génération se limite à une tâche morphologique en raison du travail important qui est déjà fournit par l'étape de transfert sur la langue cible. Les responsables du système ont donc fait un effort à ce niveau lors des dernières années afin de disposer d'un module de génération plus développé qui reste quasiment indépendant du langage source. Ces progrès permettent d'être optimiste en ce qui concerne le développement d'un système Metal multilingue.

Un point qui n'a pas été abordé dans la description du système mais qui est important au niveau des possibilités de développement est l'existence d'un outil (appelé Intercoder) facilitant le travail des linguistes. Il permet de visualiser le résultat de l'analyse réalisée par le système et

d'identifier facilement quelle règle est à l'origine de la construction d'une certaine partie de la structure.

Il est difficile d'affirmer qu'un système se distingue entre Metal et Logos. Bien qu'ils aient plusieurs points communs (la gestion complète du format, la forte modularité du point de vue informatique et linguistique, ...), ils ont tous deux des caractéristiques propres qui représentent des avantages non négligeables.

De son côté, Metal repose sur des bases linguistiques plus solides, Hutchins prétend même que : « *Despite some reservations, the Metal German-English system represents the most linguistically and computationally advanced MT system available at present on the commercial market* » [HUTCHINS]. Si on ajoute à cet avantage le fait que les linguistes disposent d'un outil (mais pas les clients, au contraire de Logos) facilitant fortement la compréhension de la manière de fonctionner du système, on est en droit de penser que Metal dispose encore d'une importante marge de progression.

Par contre, nous avons vu que Metal ne fait référence à aucune caractéristique sémantique lors de l'étape d'analyse. Metal a toujours été très prudent en ce qui concerne l'emploi des informations sémantiques, même lors de l'étape de transfert. De ce point de vue, Logos dispose donc d'un avantage important car beaucoup d'ambiguïtés ne peuvent être résolues qu'au niveau sémantique.

En conclusion, le système Metal représente certainement une des valeurs sûres de la Traduction Automatique, principalement s'ils parviennent à donner une plus grande importance à la sémantique dans leur système

Conclusion

La première partie de ce mémoire a introduit le lecteur non initié aux principes de base qui constituent les fondements de la linguistique et de la Traduction Automatique. Pour cela, nous avons consacré le premier chapitre à la description des niveaux de connaissance linguistique présents dans le langage ainsi que des modèles linguistiques les plus souvent utilisés dans le cadre de la Traduction Automatique. Quant au second chapitre, il a permis de situer le domaine de la Traduction Automatique par l'intermédiaire d'un bref historique et d'une présentation des caractéristiques propres à ce domaine. Ce chapitre a été clôturé par une brève discussion sur les rapports entre la Traduction Automatique et l'Intelligence Artificielle.

La seconde partie a été entièrement consacrée au système Logos. Dans le chapitre III, l'architecture du système a été décrite en détail afin de fournir au lecteur tous les éléments nécessaires à la compréhension des critiques que nous avons formulées dans le chapitre IV. L'étude du système Logos a montré que ce système dispose d'un grand potentiel mais que la méthode suivie au cours du développement le rend fragile. Le principe du *pattern matching*, qui se marie parfaitement avec la taxonomie offerte par les *SAL codes*, représente la clé de voûte sur laquelle repose tout le système. En effet, cela a permis de créer un module de résolution des homographes (Res) dont la qualité des résultats connaît peu d'équivalents, et un module de transfert (Tran1,2,3 et 4) dont les règles peuvent traiter des cas très spécifiques sans interférer avec d'autres règles. De plus, bien que les résultats obtenus soient déjà très satisfaisant, nous avons montré qu'il reste encore une marge de progression non négligeable.

Malheureusement, il manque aux règles de Logos des fondements théoriques plus solides permettant une gestion satisfaisante de la complexité du langage naturel. L'étude des différents systèmes dans la troisième partie a permis de constater que cette absence de théorie est typique des systèmes commerciaux. Seuls les systèmes académiques tels que Eurotra et Susy, qui ont été développés par des universités, tentent de baser leurs recherches sur des théories linguistiques. Cependant, force est de constater que ces systèmes ont beaucoup de difficultés à atteindre l'efficacité des systèmes commerciaux tels que Systran et Logos. Le seul système qui semble reposer sur des bases théoriques valables (en raison de son développement initial en Université) mais qui est maintenant de type commercial est le système Metal. Cependant, nous avons vu qu'il lui reste beaucoup de progrès à faire au niveau de la sémantique.

Tous les systèmes que nous avons analysés sont de type *general purpose* (à l'exception de TAUM-Meteo). On a constaté que cette optique dirige tous les systèmes vers une approche de type transfert, et donc de type indirecte. Nous avions déjà évoqué dans le second chapitre les insuffisances évidentes d'une approche directe, ce qui nous semble important à souligner ici est l'abandon généralisé de l'approche *interlingua* malgré les nombreux avantages qu'elle comporte (voir Metal).

Logos reste relativement méconnu dans le domaine de la Traduction Automatique. Ce phénomène est dû à la volonté des responsables de Logos de ne pas dévoiler les principes de leur système par souci de protectionnisme. Mais la compagnie a décidé de modifier cette politique et de faire connaître les avantages de Logos afin de démontrer qu'il s'agit d'un des meilleurs systèmes de sa génération. C'est dans le cadre de cette politique d'ouverture que s'inscrit notre travail. Le chapitre consacré à l'explication du système Logos a permis au lecteur de comprendre les principes de base sur lesquels repose le système. Nous avons ensuite montré la position avantageuse dont dispose Logos au sein des systèmes les plus connus et les plus performants dans le domaine de la Traduction Automatique.

Références bibliographiques

[ALLEGRANZA] V. Allegranza, S. Krauwer, E. Steiner, *Eurotra Special Issue*, in Machine Translation 6, N°.2/3, 1991.

[ANANIADOU] E. Ananiadou, S. Warwick, *An Overview of Post-ALPAC Developments*, Paper presented for the Lugano Tutorial on Machine Translation April 2-6th, 1984.

[BARR] A. Barr et E. A. Feigenbaum, *The Handbook of Artificial Intelligence*, volume I, Chapter IV : *Understanding Natural Language*, Pitman, 1981.

[BERLEUR] J. Berleur s.j., *Ordinateurs et Langages : des "Novlangues" à l'Horizon ?*, Enjeux N°. 12, mai 1987.

[CEUSTERS] W. Ceusters, G. Deville, E. Herbigniaux, P. Mousel, O. Streiter, G. Thienpont, *The ANTHEM Prototype*, IAI Working Papers N°. 31, september 1994.

[DEVILLE] G. Deville, *Modelization of Task-Oriented Utterances in a Man-Machine Dialogue System*, Thèse de Doctorat, 1989.

[GDANIEC_a] C. Gdaniec, *Input-Text Identification with an Eye to MT : A Translatability Index*, AMTA Workshop on "text and MT", Philadelphia, October 5, 1993.

[GDANIEC_b] C. Gdaniec, *The Logos Translatability Index*, Document interne.

[HUTCHINS] W. J. Hutchins et H. L. Somers, *An introduction to Machine Translation*, Academic Press, 1992.

[ISABELLE] P. Isabelle, *Machine Translation at the TAUM Group*, (Référence incomplète).

[LAROUSSE] *Le Petit Larousse Illustré*, grand format, Larousse, 1995.

[MAAS_a] H.-D. Maas, *The MT System SUSY*, Paper presented for the Lugano Tutorial on Machine Translation April 2-6th, 1984.

[MAAS_b] H.-D. Maas, *The MT System SUSY,* In King, 1987, pp. 209-246.

[MAEGAARD_a] B. Maegaard, H. Ruus, *Structuring Linguistic Information for Machine Translation, The Eurotra Interface Structure* (Référence incomplète).

[MAEGAARD_b] B. Maegaard, H. Ruus, *An Introduction to Eurotra,* in Bulletin Terminologie N°. 40, 1981.

[MANUAL] *Logos, The Intelligent Translation System,* Reference Manual for English Source, Version 7.7.

[MINSKY] M. Minsky, *A Framework for Representing Knowledge,* Memo 306, MIT, Cambridge Mass. in *Readings in knowledge representation,* Brachman & Levesque, Morgan Kaufmann publishers, Inc, Los Altos, pp. 245-262, 1985.

[PIGGOT] I. M. Piggot, *How does Systran translate?*, Commission of the European Communities, Luxembourg, 1981.

[SABAH_a] G. Sabah, *L'Intelligence et le Langage, Volume 1 : Représentation des Connaissances*, 2° édition, Hermès 1990.

[SABAH_b] G. Sabah, *L'Intelligence et le Langage, Volume 2 : Processus de Compréhension*, 2° édition, Hermès, 1990.

[SCOTT_a] B. E. Scott, *The Logos system*, paper delivered at the MT SUMMIT II Conference in Munich, august 16, 1989.

[SCOTT_b] B. E. Scott, *An Overview of the Architecture, Representation, and Process of the Logos System*, paper prepared for Union Fenosa, 1993.

[SCOTT_c] B. E. Scott, *Competence, Performance and the Paradigm Shift : a Connectionist Perspective.*

[SCOTT_d] B. E. Scott, *Optimizer & Logos Integrate for Quality Translation*, in multilingual computing, Volume 5, Issue 3, pp. 28-31, 1994.

[SCHANK] R. Schank, R. Abelson, *Scripts, plans, goals and understanding*, Hillsdale, N.J. Lawrence Erlbaum, 1977.

[SHARP_a] R. Sharp, *CAT2 - Implementing a Formalism for Multi-Lingual MT*, Proceedings of the second International Conference on Theoretical & Methodological Issues in Machine Translation of Natural Language, Pittsburg, PA, 1988.

[SHARP_b] R. Sharp, *CAT2 Reference Manual Version 3.6*, IAI Working Papers N°.27, august 1994.

[TUCKER] A. B. Tucker Jr., S. Nirenburg, *Machine Translation : A Contemporary View*, Annual Review of Information Science and Technology (ARIST), Vomule 19, 1984, Martha Williams, Editor.

[WILKS] Y. Wilks, *Machine Translation and Artificial Intelligence*, in Translating and the Computer, B.M. Snell, 1979.

Annexe :

Rapport sur le Translatability Index

CONTENTS

1. training set : 3 files
   tesf set : 10 files
   sentences : all
   QPI distrib. : natural
   Factors : all

I. Varying the number of sentences

2. training set : 3 files
   tesf set : 7 files
   sentences : max 49 sentences or each QPI
   QPI distrib. : natural
   Factors : all

II. Varying the number of files in the training set

3. training set : 8 files
   tesf set : 2 files
   sentences : all
   QPI distrib. : natural
   Factors : all

III. Varying the QPI distribution

4. training set : 3 files
   tesf set : 10 files
   sentences : all
   QPI distrib. : no qpi<3
   Factors : all

5. training set : 3 files
   tesf set : 10 files
   sentences : all
   QPI distrib. : no qpi<4
   Factors : all

6. training set : 3 files
   tesf set : 7 files
   sentences : all
   QPI distrib. : no qpi>5
   Factors : all

IV. Varying the number of factors

7. training set : 3 files
   tesf set : 10 files
   sentences : all
   QPI distrib. : natural
   Factors : 5

8. training set : 3 files
   tesf set : 10 files
   sentences : all
   QPI distrib. : natural
   Factors : 10

V. "all but 1 file in the training set" tests

9. training set : 12 files
   tesf set : 1 file
   sentences : all
   QPI distrib. : natural
   Factors : all

10. training set : 12 files
    tesf set : 1 file
    sentences : no sentences with less than 4 for words
    QPI distrib. : natural
    Factors : all

11. training set : 12 files
    tesf set : 1 file
    sentences : no sentences with less than 3 for words
    QPI distrib. : natural
    Factors : all

VI. Detailed results about 3 files

Type of calculation :

The calculations are made with all sentences of the training set and with all factors.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258

Test set :

amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32
clin2test.stattix	sentence number :	176
dage7test.stattix	sentence number :	414
papert.stattix	sentence number :	78

File 1 : engnote1.stattix sentence number : 339
File 2 : microsof.stattix sentence number : 119
File 3 : nordson.stattix sentence number : 258

RESULTS OF THE WEIGHTS CALCULATION
==========

constant 2.048761

Fully Common :

1	Number of the Sentence	factor excluded by the user
2	Qpi	factor excluded by the user
3	Length	factor excluded by the user
4	Broken Sentences	no occurrence
5	Short Parens	-0.463829
6	Long Parens	0.172505
7	Coord. Conj	0.117881
8	Homographs	0.104834
9	No Verb	0.207775
10	All Upper Case	-0.899368
11	Source language	always the same value
12	Marked Interrog	0.147500
13	WH Interrog	0.374411
14	Suspicious EOS	-1.522878
15	SWORK to Verb	0.023249
16	Dble-quoted Verb SWORK	0.423098
17	Unmatched Left Parenth	1.166472
18	Not WC20 Count	-0.019346
19	All Initial Caps	-0.514378

Mixed :

1	Dependent Clause	0.061042
2	DC Secondary	0.266193
3	Nested Parens in REL	-0.747054
4	REL Clauses	0.417832
5	Commas	0.013672
6	Tagged Interrog	1.592942
7	TH Clauses	0.245958
8	Unfound Words	0.184839

Source Specific :

1	No Main Clause	singular matrix
2	Compound Clause	0.056757
3	INGs in Sent.	0.272605
4	ING Clause	0.117481
5	AS	0.130095
6	BOS AS	-0.175738
7	WITH	0.007532
8	Long NP	0.033465

```
 9                    ING NP       0.168494
10          Quotes/Bold/Etc.    no occurrence
11        Initial CapStrings       0.468202
12    All Upper Case_Strings       0.158941
13             Does not exist    factor excluded by the user
14             Does not exist    factor excluded by the user
15             Does not exist    factor excluded by the user
16             Does not exist    factor excluded by the user

Number of factors taken into account          : 33/41
Total number of sentences taken into account  : 637/716
```

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	2	4.951363	-4.951363	0.510350	82.522722 %
2	49	2.554203	-2.554203	0.614444	42.570058 %
3	68	1.478809	-1.450939	0.633817	24.646810 %
4	96	0.809597	-0.741783	0.441570	13.493276 %
5	154	0.453634	0.022724	0.457765	7.560563 %
6	152	0.893834	0.882224	0.428954	14.897233 %
7	116	1.442553	1.442553	0.366211	24.042554 %
	637	1.077547	-0.000000	1.077547	17.959118 %

N.B. : all differences are calculated by doing TI-QPI

Significant factors

```
CO  0                  Constant 0.000000
FC  8                Homographs 0.000336
MX  4             REL Clauses   0.001360
FC 10           All Upper Case   0.030028
FC  5             Short Parens   0.046273
MX  8           Unfound Words    0.086839
SS  3           INGs in Sent.    0.128104
MX  6         Tagged Interrog    0.128826
FC 19        All Initial Caps    0.134438
MX  7               TH Clauses   0.168410
FC 17  Unmatched Left Parenth    0.246273
FC  7               Coord. Conj  0.247239
FC 14          Suspicious EOS    0.272975
FC 18          Not WC20 Count    0.273357
MX  3    Nested Parens in REL    0.278386
```

Ideal set of factors :

```
CO  0                  Constant
FC  8                Homographs
MX  4             REL Clauses
```

Filename : amdahl.stattix

Average QPI for the document : 5.758621
Average TI for the document : 5.198951
Average difference between TI and QPI : -0.559670
Total number of sentences taken into account : 116/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	3.354193	3.354193	0.000000	55.903217 %
3	6	2.126128	2.126128	0.448270	35.435464 %
4	9	0.733005	0.601641	0.517802	12.216746 %
5	26	0.652473	-0.048041	0.667254	10.874545 %
6	36	0.822948	-0.807157	0.389316	13.715805 %
7	38	1.477388	-1.477388	0.392832	24.623137 %
	116	1.081372	-0.559670	0.891729	18.022871 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.141509
Average TI for the document : 5.301484
Average difference between TI and QPI : -0.840026
Total number of sentences taken into account : 106/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	2	1.931547	1.931547	0.495055	32.192458 %
4	9	0.900111	0.633682	0.609673	15.001856 %
5	12	0.491869	-0.101072	0.491869	8.197818 %
6	32	1.126686	-1.126686	0.492991	18.778107 %
7	51	1.202787	-1.202787	0.574414	20.046450 %
	106	1.087383	-0.840026	0.700376	18.123055 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 5.027778
Average TI for the document : 4.896385
Average difference between TI and QPI : -0.131393
Total number of sentences taken into account : 36/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	2.576853	2.576853	0.599777	42.947542 %
3	7	1.207733	0.996288	0.891017	20.128888 %
4	2	1.292951	1.292951	0.336795	21.549183 %
5	6	0.314780	-0.298123	0.294131	5.246328 %
6	5	0.952610	-0.952610	0.553804	15.876840 %
7	12	1.503806	-1.503806	0.244003	25.063431 %
	36	1.279023	-0.131393	1.243923	21.317058 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.500000
Average TI for the document : 4.560311
Average difference between TI and QPI : 0.060311
Total number of sentences taken into account : 18/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.277188	5.277188	0.000000	87.953133 %
2	2	2.305805	2.305805	0.276029	38.430075 %
3	5	0.911359	0.799107	0.759461	15.189313 %
4	1	0.112560	0.112560	0.000000	1.876000 %
5	0	NaN	NaN	NaN	NaN %
6	6	1.251822	-1.251822	0.869745	20.863694 %
7	3	1.800121	-1.800121	0.643214	30.002017 %
	18	1.526080	0.060311	1.532781	25.434670 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.083601
Average TI for the document : 5.028605
Average difference between TI and QPI : -0.054996
Total number of sentences taken into account : 311/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.441011	4.441011	0.000000	74.016850 %
2	20	2.616257	2.616257	0.792958	43.604287 %
3	31	1.451875	1.414125	0.676673	24.197909 %
4	58	0.773321	0.684252	0.474405	12.888683 %
5	65	0.437530	0.012624	0.434616	7.292163 %
6	62	0.876115	-0.862404	0.427613	14.601919 %
7	74	1.415486	-1.415486	0.374234	23.591441 %
	311	1.074378	-0.054996	1.069965	17.906294 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.560311
Average TI for the document : 5.017137
Average difference between TI and QPI : -0.543175
Total number of sentences taken into account : 257/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.690872	4.690872	0.000000	78.181200 %
2	8	2.425007	2.425007	0.680440	40.416785 %
3	15	1.259711	1.259711	0.500063	20.995178 %
4	24	0.845711	0.608174	0.588202	14.095176 %
5	43	0.522700	-0.361861	0.475122	8.711662 %
6	106	0.976984	-0.968180	0.438987	16.283063 %
7	60	1.316527	-1.316527	0.414301	21.942109 %
	257	1.044014	-0.543175	0.818622	17.400234 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.125000
Average TI for the document : 4.729372
Average difference between TI and QPI : 0.604372
Total number of sentences taken into account : 32/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.379086	2.379086	0.847557	39.651430 %
3	9	1.550873	1.550873	0.634967	25.847880 %
4	1	0.880755	0.880755	0.000000	14.679250 %
5	13	0.628698	-0.230990	0.666964	10.478306 %
6	2	0.829752	-0.829752	0.177172	13.829208 %
7	2	1.365885	-1.365885	0.843486	22.764758 %
	32	1.228075	0.604372	1.128316	20.467914 %

rem : all differences are calculated by doing TI-QPI

Filename : clin2test.stattix

Average QPI for the document : 5.222222
Average TI for the document : 4.820059
Average difference between TI and QPI : -0.402163
Total number of sentences taken into account : 162/176

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	2	1.577784	1.577784	0.032885	26.296392 %
3	18	1.613284	1.409440	0.735386	26.888065 %
4	23	0.507182	0.171163	0.467802	8.453032 %
5	41	0.695643	-0.403382	0.683334	11.594057 %
6	55	1.182309	-1.182309	0.426699	19.705148 %
7	23	0.973260	-0.697696	0.854682	16.221002 %
	162	0.986378	-0.402163	0.894550	16.439631 %

rem : all differences are calculated by doing TI-QPI

Filename : dage7test.stattix

Average QPI for the document : 5.583960
Average TI for the document : 4.979679
Average difference between TI and QPI : -0.604281
Total number of sentences taken into account : 399/414

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.974627	2.530764	1.456169	49.577123 %
3	23	1.525339	1.497133	0.638432	25.422311 %
4	53	0.695707	0.573613	0.513976	11.595118 %
5	73	0.446870	-0.227916	0.441412	7.447827 %
6	143	1.059216	-0.990172	0.497447	17.653607 %
7	102	1.647635	-1.572206	0.556240	27.460591 %
	399	1.100192	-0.604281	0.925624	18.336535 %

rem : all differences are calculated by doing TI-QPI

Filename : papert.stattix

Average QPI for the document : 4.445946
Average TI for the document : 4.937839
Average difference between TI and QPI : 0.491893
Total number of sentences taken into account : 74/78

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	6	1.694322	1.694322	0.740014	28.238697 %
3	15	1.664394	1.664394	0.559812	27.739892 %
4	13	0.991446	0.984836	0.409100	16.524097 %
5	23	0.401189	0.069119	0.398956	6.686480 %
6	14	0.665915	-0.665915	0.364020	11.098585 %
7	3	1.267188	-1.267188	0.114897	21.119806 %
	74	0.950978	0.491893	0.907726	15.849628 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

The calculations are made with 49 sentences for each of the possible values for the qpi (except for qpi=1) and with all factors.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258

Test set :

amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32

```
File   1 :     engnote1.stattix      sentence number :  339
File   2 :     microsof.stattix      sentence number :  119
File   3 :      nordson.stattix      sentence number :  258

RESULTS OF THE WEIGHTS CALCULATION
==================================

                    constant      2.625086

Fully Common :
------------
 1    Number of the Sentence   factor excluded by the user
 2                       Qpi   factor excluded by the user
 3                    Length   factor excluded by the user
 4          Broken Sentences   no occurrence
 5              Short Parens     -1.269257
 6               Long Parens      0.044158
 7               Coord. Conj      0.206680
 8                Homographs      0.107198
 9                   No Verb     -0.466659
10            All Upper Case     -1.136123
11          Source language    always the same value
12          Marked Interrog      0.133999
13              WH Interrog      0.340063
14           Suspicious EOS    no occurrence
15            SWORK to Verb      0.006474
16    Dble-quoted Verb SWORK     0.147970
17    Unmatched Left Parenth     0.894365
18           Not WC20 Count     -0.040106
19         All Initial Caps      0.606542

Mixed :
-----
 1          Dependent Clause      0.252373
 2             DC Secondary     -0.443793
 3      Nested Parens in REL     -1.025717
 4              REL Clauses      0.602434
 5                   Commas      0.139443
 6           Tagged Interrog      1.163759
 7                TH Clauses      0.202455
 8            Unfound Words      0.199536

Source Specific :
---------------
 1            No Main Clause   singular matrix
 2          Compound Clause     -0.042023
 3            INGs in Sent.      0.561525
 4               ING Clause      0.093315
 5                       AS      0.085942
 6                   BOS AS     -0.217504
 7                     WITH      0.200413
 8                  Long NP      0.345755
 9                   ING NP      0.150026
```

```
10          Quotes/Bold/Etc.   no occurrence
11        Initial CapStrings   no occurrence
12    All Upper Case_Strings   no occurrence
13             Does not exist  factor excluded by the user
14             Does not exist  factor excluded by the user
15             Does not exist  factor excluded by the user
16             Does not exist  factor excluded by the user
```

Number of factors taken into account : 30/41
Total number of sentences taken into account : 296/716

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	2	3.920228	-3.920228	0.118223	65.337125 %
2	49	1.982964	-1.948691	0.716823	33.049393 %
3	49	1.040262	-0.865978	0.711972	17.337704 %
4	49	0.553790	-0.249562	0.525897	9.229825 %
5	49	0.562980	0.340474	0.550245	9.383006 %
6	49	1.151484	1.104956	0.497298	19.191398 %
7	49	1.781174	1.778810	0.456472	29.686227 %
	296	1.197299	-0.000000	1.197299	19.954981 %

N.B. : all differences are calculated by doing TI-QPI

Significant factors

```
CO  0                 Constant 0.000000
MX  4            REL Clauses   0.002669
FC  5           Short Parens   0.007188
FC  8             Homographs   0.027478
SS  3           INGs in Sent.  0.049072
FC 10          All Upper Case  0.055756
SS  8                Long NP   0.080091
FC 18          Not WC20 Count  0.152941
MX  5                Commas    0.204523
FC  7           Coord. Conj    0.217531
FC  9               No Verb    0.237567
MX  8           Unfound Words  0.256439
MX  3      Nested Parens in REL 0.299199
```

Ideal set of factors :

```
CO  0                 Constant
MX  4            REL Clauses
```

Filename : amdahl.stattix

Average QPI for the document : 5.758621
Average TI for the document : 4.636889
Average difference between TI and QPI : -1.121732
Total number of sentences taken into account : 116/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	2.761788	2.761788	0.000000	46.029800 %
3	6	1.453752	1.453752	0.640401	24.229192 %
4	9	0.636954	-0.169000	0.655732	10.615902 %
5	26	0.890495	-0.656133	0.865907	14.841585 %
6	36	1.388830	-1.388830	0.438389	23.147163 %
7	38	1.942340	-1.921759	0.525575	32.372333 %
	116	1.415315	-1.121732	0.915074	23.588584 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.141509
Average TI for the document : 4.768804
Average difference between TI and QPI : -1.372705
Total number of sentences taken into account : 106/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	2	1.212549	1.212549	0.860709	20.209150 %
4	9	0.690556	-0.110998	0.702484	11.509269 %
5	12	0.782089	-0.646127	0.595976	13.034819 %
6	32	1.869244	-1.869244	0.700879	31.154060 %
7	51	1.558243	-1.556148	0.899769	25.970712 %
	106	1.484069	-1.372705	0.950324	24.734489 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 5.027778
Average TI for the document : 4.430306
Average difference between TI and QPI : -0.597472
Total number of sentences taken into account : 36/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	1.898647	1.898647	0.904371	31.644108 %
3	7	1.166985	0.195022	1.139125	19.449752 %
4	2	1.032647	1.032647	0.356725	17.210783 %
5	6	0.969179	-0.969179	0.494131	16.152981 %
6	5	1.452995	-1.452995	0.686618	24.216583 %
7	12	1.621163	-1.621163	0.521041	27.019390 %
	36	1.398966	-0.597472	1.177351	23.316104 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.500000
Average TI for the document : 3.380361
Average difference between TI and QPI : -1.119639
Total number of sentences taken into account : 18/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.275137	4.275137	0.000000	71.252283 %
2	2	0.849843	0.849843	0.579691	14.164050 %
3	5	1.410054	-0.652808	1.540615	23.500897 %
4	1	1.434734	-1.434734	0.000000	23.912233 %
5	0	NaN	NaN	NaN	NaN %
6	6	2.335525	-2.335525	1.378467	38.925419 %
7	3	2.472135	-2.472135	0.726144	41.202256 %
	18	1.993855	-1.119639	1.732019	33.230910 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.083601
Average TI for the document : 4.628919
Average difference between TI and QPI : -0.454682
Total number of sentences taken into account : 311/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.038447	4.038447	0.000000	67.307450 %
2	20	2.274090	2.231764	0.816224	37.901500 %
3	31	0.994063	0.943666	0.669315	16.567722 %
4	58	0.557971	0.219303	0.536180	9.299511 %
5	65	0.596410	-0.370352	0.565320	9.940159 %
6	62	1.273578	-1.236806	0.537192	21.226297 %
7	74	1.775864	-1.774299	0.454963	29.597740 %
	311	1.163476	-0.454682	1.096828	19.391260 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.560311
Average TI for the document : 4.411341
Average difference between TI and QPI : -1.148970
Total number of sentences taken into account : 257/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.207270	4.207270	0.000000	70.121167 %
2	8	1.716657	1.411761	0.860851	28.610950 %
3	15	0.795879	0.626502	0.686498	13.264652 %
4	24	0.663943	-0.066207	0.680494	11.065713 %
5	43	1.094853	-0.992322	0.656484	18.247550 %
6	106	1.487487	-1.483814	0.635606	24.791448 %
7	60	1.977352	-1.977352	0.365588	32.955872 %
	257	1.436602	-1.148970	0.891127	23.943370 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.125000
Average TI for the document : 4.157535
Average difference between TI and QPI : 0.032535
Total number of sentences taken into account : 32/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	1.905455	1.905455	1.146221	31.757580 %
3	9	1.318729	0.919319	1.067532	21.978809 %
4	1	0.464836	0.464836	0.000000	7.747267 %
5	13	0.979150	-0.925472	0.812387	16.319159 %
6	2	1.058048	-1.058048	0.122735	17.634133 %
7	2	1.538805	-1.538805	0.842264	25.646750 %
	32	1.243229	0.032535	1.245262	20.720478 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

The calculations are made with all sentences of the training set and with all factors.

Training set :

amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
engnote1.stattix	sentence number :	339
infobull.stattix	sentence number :	269
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258

Test set :

capoten1.stattix	sentence number :	36
schneider.stattix	sentence number :	32

File	1 :	amdahl.stattix	sentence number :	125
File	2 :	berlitz.stattix	sentence number :	106
File	3 :	comtec.stattix	sentence number :	21
File	4 :	consumer.stattix	sentence number :	76
File	5 :	engnote1.stattix	sentence number :	339
File	6 :	infobull.stattix	sentence number :	269
File	7 :	microsof.stattix	sentence number :	119
File	8 :	nordson.stattix	sentence number :	258

RESULTS OF THE WEIGHTS CALCULATION
==========

	constant	1.919707

Fully Common :

1	Number of the Sentence	factor excluded by the user
2	Qpi	factor excluded by the user
3	Length	factor excluded by the user
4	Broken Sentences	no occurrence
5	Short Parens	-0.311603
6	Long Parens	-0.078235
7	Coord. Conj	-0.040004
8	Homographs	0.076142
9	No Verb	-0.027959
10	All Upper Case	-0.765625
11	Source language	always the same value
12	Marked Interrog	0.523485
13	WH Interrog	-0.181528
14	Suspicious EOS	-1.248151
15	SWORK to Verb	0.001107
16	Dble-quoted Verb SWORK	0.787702
17	Unmatched Left Parenth	1.266236
18	Not WC20 Count	0.006576
19	All Initial Caps	-0.415790

Mixed :

1	Dependent Clause	0.054209
2	DC Secondary	0.204358
3	Nested Parens in REL	-0.495596
4	REL Clauses	0.245406
5	Commas	-0.026418
6	Tagged Interrog	1.422493
7	TH Clauses	0.132379
8	Unfound Words	0.341173

Source Specific :

1	No Main Clause	singular matrix
2	Compound Clause	0.047160
3	INGs in Sent.	0.152401
4	ING Clause	0.048572

5	AS	0.096449	
6	BOS AS	0.468432	
7	WITH	0.192842	
8	Long NP	-0.013985	
9	ING NP	0.238871	
10	Quotes/Bold/Etc.	no occurrence	
11	Initial CapStrings	0.876528	
12	All Upper Case_Strings	0.305648	
13	Does not exist	factor excluded by the user	
14	Does not exist	factor excluded by the user	
15	Does not exist	factor excluded by the user	
16	Does not exist	factor excluded by the user	

Number of factors taken into account : 33/41
Total number of sentences taken into account : 1119/1313

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	2	5.086399	-5.086399	0.402002	84.773321 %
2	67	2.769552	-2.769552	0.551559	46.159194 %
3	104	1.736091	-1.716611	0.567681	28.934847 %
4	151	0.933166	-0.914025	0.433987	15.552772 %
5	250	0.447191	-0.078407	0.434410	7.453181 %
6	337	0.768671	0.758071	0.382593	12.811184 %
7	208	1.328894	1.328894	0.356660	22.148227 %
	1119	1.040611	0.000000	1.040611	17.343522 %

N.B. : all differences are calculated by doing TI-QPI

Significant factors

CO	0	Constant	0.000000
FC	8	Homographs	0.000142
MX	8	Unfound Words	0.000176
MX	4	REL Clauses	0.004667
FC	12	Marked Interrog	0.030176
FC	10	All Upper Case	0.031338
FC	5	Short Parens	0.058672
MX	6	Tagged Interrog	0.144748
FC	19	All Initial Caps	0.153419
FC	16	Dble-quoted Verb SWORK	0.155198
SS	3	INGs in Sent.	0.169310
FC	17	Unmatched Left Parenth	0.182642
SS	7	WITH	0.205804
SS	9	ING NP	0.250583
SS	11	Initial CapStrings	0.255443
MX	7	TH Clauses	0.273671
MX	3	Nested Parens in REL	0.273792

Ideal set of factors :

CO	0	Constant
FC	8	Homographs

MX 8	Unfound Words
MX 4	REL Clauses
FC 12	Marked Interrog

Filename : capoten1.stattix

Average QPI for the document : 5.027778
Average TI for the document : 4.961009
Average difference between TI and QPI : -0.066769
Total number of sentences taken into account : 36/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	2.820169	2.820169	0.599039	47.002813 %
3	7	0.975164	0.975164	0.526251	16.252733 %
4	2	0.798859	0.798859	0.615342	13.314325 %
5	6	0.529561	-0.479904	0.415984	8.826017 %
6	5	0.892233	-0.892233	0.612601	14.870557 %
7	12	1.230635	-1.230635	0.235843	20.510589 %
	36	1.169742	-0.066769	1.154904	19.495694 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.125000
Average TI for the document : 5.006637
Average difference between TI and QPI : 0.881637
Total number of sentences taken into account : 32/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.765714	2.765714	0.823746	46.095227 %
3	9	1.956860	1.956860	0.481989	32.614330 %
4	1	1.068756	1.068756	0.000000	17.812600 %
5	13	0.619292	-0.001336	0.619600	10.321536 %
6	2	0.693017	-0.693017	0.185671	11.550283 %
7	2	1.446639	-1.446639	0.945212	24.110650 %
	32	1.401224	0.881637	1.209786	23.353735 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

For training and testing sets, no sentences with qpi<3 were included.

All factors are used.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258

Test set :

amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32
clin2test.stattix	sentence number :	176
dage7test.stattix	sentence number :	414
papert.stattix	sentence number :	78

File 1 : engnote1.stattix sentence number : 339
File 2 : microsof.stattix sentence number : 119
File 3 : nordson.stattix sentence number : 258

RESULTS OF THE WEIGHTS CALCULATION
==========

constant 1.806295

Fully Common :

1	Number of the Sentence	factor excluded by the user
2	Qpi	factor excluded by the user
3	Length	factor excluded by the user
4	Broken Sentences	no occurrence
5	Short Parens	-0.188286
6	Long Parens	0.227867
7	Coord. Conj	0.055308
8	Homographs	0.070965
9	No Verb	0.332136
10	All Upper Case	-1.040651
11	Source language	always the same value
12	Marked Interrog	0.089839
13	WH Interrog	0.266606
14	Suspicious EOS	-1.292163
15	SWORK to Verb	0.035427
16	Dble-quoted Verb SWORK	-0.349935
17	Unmatched Left Parenth	1.295345
18	Not WC20 Count	0.001320
19	All Initial Caps	-0.814102

Mixed :

1	Dependent Clause	0.078289
2	DC Secondary	0.465854
3	Nested Parens in REL	-0.389190
4	REL Clauses	0.232232
5	Commas	0.006626
6	Tagged Interrog	1.457252
7	TH Clauses	0.184522
8	Unfound Words	0.014402

Source Specific :

1	No Main Clause	suppressed because of singular matrix
2	Compound Clause	0.149622
3	INGs in Sent.	0.364255
4	ING Clause	0.080414
5	AS	0.187538
6	BOS AS	0.108326
7	WITH	-0.011264
8	Long NP	-0.044951

```
 9                    ING NP       0.151936
10          Quotes/Bold/Etc.    no occurrence
11        Initial CapStrings       0.666912
12    All Upper Case_Strings       0.337230
13             Does not exist    factor excluded by the user
14             Does not exist    factor excluded by the user
15             Does not exist    factor excluded by the user
16             Does not exist    factor excluded by the user

Number of factors taken into account          : 33/41
Total number of sentences taken into account  : 586/716
```

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	68	1.732724	1.706655	3.413310	28.878736 %
4	96	0.985185	0.961303	1.922606	16.419758 %
5	154	0.448760	0.187364	0.540993	7.479339 %
6	152	0.694135	-0.648351	1.299527	11.568914 %
7	116	1.195192	-1.195192	2.390384	19.919868 %
	586	0.897036	0.000000	0.897036	14.950595 %

N.B. : all differences are calculated by doing TI-QPI

Significant factors

```
CO  0                  Constant 0.000000
FC 10           All Upper Case  0.002931
FC  8               Homographs  0.004949
FC 19         All Initial Caps  0.005811
SS  3            INGs in Sent.  0.018290
MX  4              REL Clauses  0.040304
FC 15            SWORK to Verb  0.083220
MX  6          Tagged Interrog  0.090513
FC 17  Unmatched Left Parenth   0.115716
FC  9                  No Verb  0.130241
MX  2             DC Secondary  0.139858
SS  2          Compound Clause  0.140720
SS  5                      AS   0.231766
MX  7              TH Clauses   0.233682
FC 14          Suspicious EOS   0.253947
SS 12  All Upper Case_Strings   0.283133
```

Ideal set of factors :

```
CO  0                  Constant
```

Filename : amdahl.stattix

Average QPI for the document : 5.765217
Average TI for the document : 5.418888
Average difference between TI and QPI : -0.346329
Total number of sentences taken into account : 115/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	3.408884	3.408884	0.000000	56.814733 %
3	6	2.322371	2.322371	0.401662	38.706189 %
4	9	0.880606	0.831755	0.509073	14.676761 %
5	25	0.671470	0.141478	0.618104	11.191162 %
6	36	0.636150	-0.573638	0.357966	10.602497 %
7	38	1.251125	-1.251125	0.393512	20.852087 %
	115	0.978256	-0.346329	0.874326	16.304269 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.133333
Average TI for the document : 5.529198
Average difference between TI and QPI : -0.604136
Total number of sentences taken into account : 105/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	2	2.412917	2.412917	0.236836	40.215292 %
4	9	1.025059	0.923951	0.420974	17.084313 %
5	12	0.449280	0.056181	0.446954	7.487993 %
6	32	0.903521	-0.903521	0.441204	15.058683 %
7	50	0.966743	-0.966743	0.654085	16.112380 %
	105	0.920881	-0.604136	0.733199	15.348020 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 5.000000
Average TI for the document : 5.236879
Average difference between TI and QPI : 0.236879
Total number of sentences taken into account : 35/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	2.927591	2.927591	0.451906	48.793187 %
3	7	1.579139	1.510742	0.844772	26.318976 %
4	2	1.563452	1.563452	0.242768	26.057533 %
5	6	0.307162	0.185067	0.221087	5.119372 %
6	4	0.783323	-0.783323	0.264490	13.055383 %
7	12	1.258234	-1.258234	0.225605	20.970574 %
	35	1.313323	0.236879	1.332407	21.888723 %

rem : all differences are calculated by doing TI-QPI

Filename : clin2test.stattix

Average QPI for the document : 5.222222
Average TI for the document : 5.332270
Average difference between TI and QPI : 0.110048
Total number of sentences taken into account : 162/176

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	2	1.984759	1.984759	0.303664	33.079317 %
3	18	2.006503	1.989280	0.613865	33.441709 %
4	23	0.884330	0.808446	0.464300	14.738825 %
5	41	0.585725	0.147953	0.550877	9.762079 %
6	55	0.716166	-0.716166	0.330261	11.936102 %
7	23	0.871547	-0.313914	0.840335	14.525788 %
	162	0.888121	0.110048	0.900846	14.802015 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.500000
Average TI for the document : 4.757307
Average difference between TI and QPI : 0.257307
Total number of sentences taken into account : 18/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.674351	5.674351	0.000000	94.572517 %
2	2	2.511020	2.511020	0.378247	41.850342 %
3	5	1.130223	0.977068	0.893031	18.837050 %
4	1	0.193437	0.193437	0.000000	3.223950 %
5	0	NaN	NaN	NaN	NaN %
6	6	1.108681	-1.053112	0.926865	18.478011 %
7	3	1.608323	-1.608323	0.613449	26.805383 %
	18	1.556555	0.257307	1.573719	25.942590 %

rem : all differences are calculated by doing TI-QPI

Filename : consumer.stattix

Average QPI for the document : 4.643836
Average TI for the document : 5.062048
Average difference between TI and QPI : 0.418212
Total number of sentences taken into account : 73/76

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	9	2.974036	2.974036	0.559472	49.567261 %
3	8	2.029997	2.029997	0.347438	33.833281 %
4	13	1.007944	0.981082	0.415392	16.799071 %
5	18	0.454452	0.037311	0.450039	7.574194 %
6	20	0.979741	-0.978623	0.477804	16.329022 %
7	5	1.265996	-1.265996	0.155926	21.099933 %
	73	1.235815	0.418212	1.233972	20.596920 %

rem : all differences are calculated by doing TI-QPI

Filename : dage7test.stattix

Average QPI for the document : 5.580402
Average TI for the document : 5.202053
Average difference between TI and QPI : -0.378349
Total number of sentences taken into account : 398/414

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	3.361887	2.796204	1.684164	56.031443 %
3	23	1.864272	1.858816	0.446784	31.071204 %
4	53	0.917569	0.890684	0.466232	15.292812 %
5	73	0.387130	-0.052763	0.395835	6.452174 %
6	143	0.861976	-0.773385	0.443480	14.366260 %
7	101	1.523961	-1.386902	0.538708	25.399343 %
	398	1.039603	-0.378349	0.949593	17.326712 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.080645
Average TI for the document : 5.283884
Average difference between TI and QPI : 0.203239
Total number of sentences taken into account : 310/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.718834	4.718834	0.000000	78.647233 %
2	20	3.056042	3.056042	0.681902	50.934027 %
3	31	1.754175	1.696989	0.656998	29.236247 %
4	58	0.964421	0.924889	0.456652	16.073676 %
5	65	0.433985	0.234278	0.349584	7.233077 %
6	61	0.670307	-0.635540	0.414543	11.171791 %
7	74	1.156026	-1.156026	0.396392	19.267095 %
	310	1.067094	0.203239	1.074281	17.784900 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.554688
Average TI for the document : 5.261278
Average difference between TI and QPI : -0.293410
Total number of sentences taken into account : 256/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.940103	4.940103	0.000000	82.335050 %
2	8	2.763078	2.763078	0.769675	46.051294 %
3	15	1.548551	1.548551	0.464736	25.809177 %
4	24	0.991515	0.856528	0.568204	16.525256 %
5	43	0.424467	-0.073862	0.433250	7.074455 %
6	106	0.799345	-0.764185	0.372263	13.322425 %
7	59	1.046828	-1.046828	0.492024	17.447127 %
	256	0.932871	-0.293410	0.823510	15.547845 %

rem : all differences are calculated by doing TI-QPI

Filename : microsof.stattix

Average QPI for the document : 5.522124
Average TI for the document : 5.274754
Average difference between TI and QPI : -0.247370
Total number of sentences taken into account : 113/119

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	3.035792	3.035792	0.167827	50.596529 %
3	11	1.770726	1.770726	0.476035	29.512102 %
4	8	0.739431	0.739431	0.392625	12.323852 %
5	20	0.431242	0.076479	0.409839	7.187370 %
6	39	0.774382	-0.714337	0.410879	12.906360 %
7	31	1.263219	-1.263219	0.263072	21.053651 %
	113	1.022320	-0.247370	0.955122	17.038661 %

rem : all differences are calculated by doing TI-QPI

Filename : nordson.stattix

Average QPI for the document : 4.898833
Average TI for the document : 5.345103
Average difference between TI and QPI : 0.446270
Total number of sentences taken into account : 257/258

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.971060	5.971060	0.000000	99.517667 %
2	27	2.954736	2.954736	0.566360	49.245594 %
3	27	1.737363	1.737363	0.571237	28.956058 %
4	30	1.090846	1.090846	0.346157	18.180770 %
5	72	0.495672	0.215400	0.446991	8.261196 %
6	57	0.629143	-0.585344	0.380419	10.485723 %
7	43	0.763625	-0.763625	0.525138	12.727080 %
	257	1.049683	0.446270	1.067336	17.494716 %

rem : all differences are calculated by doing TI-QPI

Filename : papert.stattix

Average QPI for the document : 4.410959
Average TI for the document : 5.195912
Average difference between TI and QPI : 0.784953
Total number of sentences taken into account : 73/78

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	6	2.350373	2.350373	0.711182	39.172886 %
3	15	1.942485	1.942485	0.444167	32.374758 %
4	13	1.257806	1.257806	0.460946	20.963433 %
5	23	0.452998	0.303301	0.369408	7.549965 %
6	14	0.515669	-0.508822	0.342861	8.594481 %
7	2	1.070904	-1.070904	0.040030	17.848408 %
	73	1.087276	0.784953	0.939569	18.121261 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.032258
Average TI for the document : 4.868560
Average difference between TI and QPI : 0.836302
Total number of sentences taken into account : 31/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.686765	2.686765	0.718820	44.779413 %
3	9	1.731773	1.731773	0.584475	28.862891 %
4	1	1.174032	1.174032	0.000000	19.567200 %
5	13	0.650735	-0.049740	0.654561	10.845583 %
6	2	0.655690	-0.655690	0.224674	10.928167 %
7	1	2.310448	-2.310448	0.000000	38.507467 %
	31	1.363716	0.836302	1.158060	22.728602 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

For training and testing sets, no sentences with qpi<4 were included.

All factors are used.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258

Test set :

amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32
clin2test.stattix	sentence number :	176
dage7test.stattix	sentence number :	414
papert.stattix	sentence number :	78

```
File   1 :     engnote1.stattix      sentence number :  339
File   2 :     microsof.stattix      sentence number :  119
File   3 :      nordson.stattix      sentence number :  258

RESULTS OF THE WEIGHTS CALCULATION
==================================

                    constant      1.687401

Fully Common :
------------
 1    Number of the Sentence   factor excluded by the user
 2                       Qpi   factor excluded by the user
 3                    Length   factor excluded by the user
 4          Broken Sentences   no occurrence
 5              Short Parens     -0.005916
 6               Long Parens      0.378284
 7               Coord. Conj     -0.047692
 8                Homographs      0.008221
 9                   No Verb      0.301972
10            All Upper Case     -0.854263
11            Source language   always the same value
12            Marked Interrog      0.271578
13                WH Interrog     -0.194262
14             Suspicious EOS     -1.049645
15              SWORK to Verb      0.018616
16    Dble-quoted Verb SWORK     -0.508955
17    Unmatched Left Parenth      1.730888
18             Not WC20 Count      0.034162
19          All Initial Caps     -0.660826

Mixed :
-----
 1          Dependent Clause      0.017139
 2               DC Secondary      0.455020
 3      Nested Parens in REL      0.030699
 4               REL Clauses      0.034322
 5                     Commas      0.019096
 6           Tagged Interrog   no occurrence
 7                TH Clauses      0.039758
 8              Unfound Words      0.057748

Source Specific :
---------------
 1             No Main Clause   suppressed because of singular
matrix
 2           Compound Clause      0.115849
 3             INGs in Sent.      0.239174
 4                ING Clause      0.198419
 5                        AS     -0.018381
 6                    BOS AS      0.549784
 7                      WITH      0.122276
 8                   Long NP     -0.091809
```

9	ING NP	0.065037
10	Quotes/Bold/Etc.	no occurrence
11	Initial CapStrings	0.729521
12	All Upper Case_Strings	0.301073
13	Does not exist	factor excluded by the user
14	Does not exist	factor excluded by the user
15	Does not exist	factor excluded by the user
16	Does not exist	factor excluded by the user

Number of factors taken into account : 32/41
Total number of sentences taken into account : 519/716

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	0	NaN	NaN	NaN	NaN %
4	96	1.242066	1.237675	2.475350	20.701092 %
5	154	0.512000	0.418759	0.855542	8.533337 %
6	152	0.469544	-0.416581	0.843886	7.825736 %
7	117	1.025517	-1.025517	2.051033	17.091943 %
	519	0.750371	0.000000	0.750371	12.506181 %

N.B. : all differences are calculated by doing TI-QPI

Significant factors

CO	0	Constant	0.000000
FC	10	All Upper Case	0.003301
FC	19	All Initial Caps	0.007394
FC	18	Not WC20 Count	0.016836
FC	17	Unmatched Left Parenth	0.067416
SS	4	ING Clause	0.080538
MX	2	DC Secondary	0.092483
FC	9	No Verb	0.104578
SS	3	INGs in Sent.	0.107240
FC	12	Marked Interrog	0.190919
FC	6	Long Parens	0.204746
SS	6	BOS AS	0.208602
SS	2	Compound Clause	0.229382
SS	8	Long NP	0.248937
FC	14	Suspicious EOS	0.263176

Ideal set of factors :

CO	0	Constant

Filename : amdahl.stattix

Average QPI for the document : 5.765217
Average TI for the document : 5.672133
Average difference between TI and QPI : -0.093084
Total number of sentences taken into account : 115/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	3.124872	3.124872	0.000000	52.081200 %
3	6	2.583986	2.583986	0.350470	43.066428 %
4	9	1.220896	1.220896	0.337027	20.348261 %
5	25	0.652784	0.456728	0.455168	10.879739 %
6	36	0.389659	-0.321743	0.283422	6.494325 %
7	38	1.056763	-1.056763	0.285208	17.612719 %
	115	0.870619	-0.093084	0.846957	14.510318 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.133333
Average TI for the document : 5.769849
Average difference between TI and QPI : -0.363484
Total number of sentences taken into account : 105/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	2	2.791643	2.791643	0.108324	46.527383 %
4	9	1.270505	1.270505	0.348089	21.175078 %
5	12	0.526118	0.392869	0.360899	8.768626 %
6	32	0.547401	-0.540786	0.277654	9.123349 %
7	50	0.851859	-0.851859	0.542445	14.197645 %
	105	0.794676	-0.363484	0.662686	13.244604 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 5.000000
Average TI for the document : 5.448655
Average difference between TI and QPI : 0.448655
Total number of sentences taken into account : 35/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	3.281511	3.281511	0.322175	54.691850 %
3	7	1.801476	1.801476	0.603304	30.024602 %
4	2	1.344116	1.344116	0.333340	22.401933 %
5	6	0.401488	0.374286	0.276726	6.691472 %
6	4	0.548201	-0.548201	0.167901	9.136692 %
7	12	1.064550	-1.064550	0.143045	17.742506 %
	35	1.308599	0.448655	1.322270	21.809976 %

rem : all differences are calculated by doing TI-QPI

Filename : clin2test.stattix

Average QPI for the document : 5.222222
Average TI for the document : 5.421523
Average difference between TI and QPI : 0.199300
Total number of sentences taken into account : 162/176

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	2	2.411897	2.411897	0.098475	40.198275 %
3	18	2.169760	2.169760	0.412120	36.162674 %
4	23	0.973681	0.968406	0.363253	16.228021 %
5	41	0.544625	0.243090	0.472601	9.077087 %
6	55	0.635850	-0.635850	0.287763	10.597494 %
7	23	0.774239	-0.385266	0.711185	12.903980 %
	162	0.872735	0.199300	0.886338	14.545577 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.500000
Average TI for the document : 5.221248
Average difference between TI and QPI : 0.721248
Total number of sentences taken into account : 18/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.637291	5.637291	0.000000	93.954850 %
2	2	3.124191	3.124191	0.308438	52.069842 %
3	5	1.630422	1.630422	0.682685	27.173707 %
4	1	0.372202	0.372202	0.000000	6.203367 %
5	0	NaN	NaN	NaN	NaN %
6	6	0.690784	-0.568600	0.608278	11.513069 %
7	3	1.338641	-1.338641	0.364450	22.310689 %
	18	1.587256	0.721248	1.600701	26.454273 %

rem : all differences are calculated by doing TI-QPI

Filename : consumer.stattix

Average QPI for the document : 4.643836
Average TI for the document : 5.318603
Average difference between TI and QPI : 0.674767
Total number of sentences taken into account : 73/76

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	9	3.271596	3.271596	0.501957	54.526604 %
3	8	2.225078	2.225078	0.366958	37.084640 %
4	13	1.181291	1.181291	0.413754	19.688188 %
5	18	0.411225	0.236547	0.380331	6.853753 %
6	20	0.606576	-0.595388	0.383952	10.109603 %
7	5	1.138771	-1.138771	0.098355	18.979510 %
	73	1.203140	0.674767	1.193884	20.052331 %

rem : all differences are calculated by doing TI-QPI

Filename : dage7test.stattix

Average QPI for the document : 5.580402
Average TI for the document : 5.479435
Average difference between TI and QPI : -0.100967
Total number of sentences taken into account : 398/414

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	3.575472	3.017905	1.764729	59.591203 %
3	23	2.234926	2.234926	0.332913	37.248764 %
4	53	1.257330	1.240681	0.390359	20.955504 %
5	73	0.383749	0.242478	0.303509	6.395813 %
6	143	0.560972	-0.480809	0.329478	9.349536 %
7	101	1.305131	-1.201771	0.429465	21.752180 %
	398	0.944648	-0.100967	0.923215	15.744139 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.080645
Average TI for the document : 5.533225
Average difference between TI and QPI : 0.452580
Total number of sentences taken into account : 310/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.040136	5.040136	0.000000	84.002267 %
2	20	3.451422	3.451422	0.543129	57.523695 %
3	31	2.162163	2.158339	0.544359	36.036046 %
4	58	1.220412	1.213145	0.428572	20.340204 %
5	65	0.518580	0.430599	0.319455	8.643006 %
6	61	0.450377	-0.411780	0.338871	7.506285 %
7	74	0.998786	-0.998786	0.327531	16.646438 %
	310	1.119259	0.452580	1.120105	18.654325 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.554688
Average TI for the document : 5.537800
Average difference between TI and QPI : -0.016887
Total number of sentences taken into account : 256/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.132672	5.132672	0.000000	85.544533 %
2	8	3.299920	3.299920	0.623922	54.998660 %
3	15	1.944608	1.944608	0.459776	32.410140 %
4	24	1.300289	1.231646	0.483992	21.671480 %
5	43	0.421470	0.305238	0.328745	7.024505 %
6	106	0.553818	-0.515020	0.308463	9.230301 %
7	59	0.900287	-0.900287	0.377801	15.004776 %
	256	0.866613	-0.016887	0.861098	14.443550 %

rem : all differences are calculated by doing TI-QPI

Filename : microsof.stattix

Average QPI for the document : 5.522124
Average TI for the document : 5.543573
Average difference between TI and QPI : 0.021449
Total number of sentences taken into account : 113/119

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	3.171600	3.171600	0.374327	52.859996 %
3	11	2.261266	2.261266	0.318411	37.687774 %
4	8	0.997860	0.997860	0.426719	16.631006 %
5	20	0.476778	0.353317	0.315611	7.946303 %
6	39	0.490179	-0.443786	0.292676	8.169656 %
7	31	1.060586	-1.060586	0.210131	17.676430 %
	113	0.947557	0.021449	0.953441	15.792615 %

rem : all differences are calculated by doing TI-QPI

Filename : nordson.stattix

Average QPI for the document : 4.898833
Average TI for the document : 5.591343
Average difference between TI and QPI : 0.692510
Total number of sentences taken into account : 257/258

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.920647	5.920647	0.000000	98.677450 %
2	27	3.330627	3.330627	0.416486	55.510456 %
3	27	2.181602	2.181602	0.422632	36.360037 %
4	30	1.349036	1.349036	0.290881	22.483933 %
5	72	0.544103	0.458388	0.362721	9.068388 %
6	57	0.444685	-0.376499	0.307721	7.411408 %
7	43	0.669540	-0.669540	0.442804	11.159007 %
	257	1.122703	0.692510	1.111673	18.711713 %

rem : all differences are calculated by doing TI-QPI

Filename : papert.stattix

Average QPI for the document : 4.410959
Average TI for the document : 5.481147
Average difference between TI and QPI : 1.070188
Total number of sentences taken into account : 73/78

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	6	3.022375	3.022375	0.497177	50.372914 %
3	15	2.220406	2.220406	0.392505	37.006763 %
4	13	1.600006	1.600006	0.351634	26.666763 %
5	23	0.589645	0.544028	0.283152	9.827416 %
6	14	0.376942	-0.335473	0.284894	6.282373 %
7	2	0.966347	-0.966347	0.040517	16.105792 %
	73	1.274139	1.070188	0.982568	21.235647 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.032258
Average TI for the document : 5.177296
Average difference between TI and QPI : 1.145038
Total number of sentences taken into account : 31/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	3.145286	3.145286	0.527506	52.421430 %
3	9	2.088627	2.088627	0.407099	34.810446 %
4	1	1.255906	1.255906	0.000000	20.931767 %
5	13	0.586134	0.195863	0.564100	9.768905 %
6	2	0.384759	-0.384759	0.203482	6.412658 %
7	1	2.060507	-2.060507	0.000000	34.341783 %
	31	1.491282	1.145038	1.200285	24.854703 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

The calculations are made with all factors but do not take in account the sentences with a qpi higher than 5.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258

Test set :

amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32

File 1 : engnote1.stattix sentence number : 339
File 2 : microsof.stattix sentence number : 119
File 3 : nordson.stattix sentence number : 258

RESULTS OF THE WEIGHTS CALCULATION
==========

constant 3.714539

Fully Common :

1	Number of the Sentence	factor excluded by the user
2	Qpi	factor excluded by the user
3	Length	factor excluded by the user
4	Broken Sentences	no occurrence
5	Short Parens	-0.535250
6	Long Parens	-0.148391
7	Coord. Conj	0.115363
8	Homographs	0.109009
9	No Verb	0.126629
10	All Upper Case	0.070167
11	Source language	always the same value
12	Marked Interrog	-0.053214
13	WH Interrog	0.478274
14	Suspicious EOS	no occurrence
15	SWORK to Verb	0.005998
16	Dble-quoted Verb SWORK	0.052775
17	Unmatched Left Parenth	0.453102
18	Not WC20 Count	-0.052084
19	All Initial Caps	1.482385

Mixed :

1	Dependent Clause	-0.039486
2	DC Secondary	-0.265325
3	Nested Parens in REL	-0.996742
4	REL Clauses	0.344005
5	Commas	-0.014780
6	Tagged Interrog	1.653034
7	TH Clauses	0.195253
8	Unfound Words	0.215349

Source Specific :

1	No Main Clause	singular matrix
2	Compound Clause	0.017548
3	INGs in Sent.	0.084724
4	ING Clause	0.016005
5	AS	0.168960
6	BOS AS	-0.613449
7	WITH	-0.211142
8	Long NP	0.182689
9	ING NP	0.019169

10	Quotes/Bold/Etc.	no occurrence
11	Initial CapStrings	-0.743680
12	All Upper Case_Strings	-0.217858
13	Does not exist	factor excluded by the user
14	Does not exist	factor excluded by the user
15	Does not exist	factor excluded by the user
16	Does not exist	factor excluded by the user

Number of factors taken into account : 32/41
Total number of sentences taken into account : 373/716

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	2	2.430383	-2.430383	0.581221	40.506379 %
2	51	1.708267	-1.708267	0.343406	28.471123 %
3	68	0.779931	-0.757966	0.341052	12.998845 %
4	96	0.252800	0.031381	0.255271	4.213330 %
5	156	0.901213	0.900715	0.273076	15.020219 %
6	0	NaN	NaN	NaN	NaN %
7	0	NaN	NaN	NaN	NaN %
	373	0.830766	0.000000	0.830766	13.846100 %

N.B. : all differences are calculated by doing TI-QPI

Significant factors

CO	0	Constant	0.000000
FC	8	Homographs	0.000096
FC	18	Not WC2O Count	0.001908
MX	4	REL Clauses	0.003901
FC	5	Short Parens	0.011924
FC	19	All Initial Caps	0.021560
MX	6	Tagged Interrog	0.044157
MX	8	Unfound Words	0.049848
MX	3	Nested Parens in REL	0.067334
SS	8	Long NP	0.084848
FC	7	Coord. Conj	0.216071
MX	7	TH Clauses	0.225992
SS	7	WITH	0.239793
SS	6	BOS AS	0.264724

Ideal set of factors :

CO	0	Constant
FC	8	Homographs

Filename : amdahl.stattix

Average QPI for the document : 5.758621
Average TI for the document : 3.924218
Average difference between TI and QPI : -1.834402
Total number of sentences taken into account : 116/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	2.847571	2.847571	0.000000	47.459517 %
3	6	1.004229	1.004229	0.266026	16.737150 %
4	9	0.377085	-0.103255	0.365613	6.284757 %
5	26	0.999760	-0.999760	0.385010	16.662659 %
6	36	2.044536	-2.044536	0.292774	34.075599 %
7	38	3.187823	-3.187823	0.472744	53.130382 %
	116	2.008630	-1.834402	1.068064	33.477159 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.141509
Average TI for the document : 3.899939
Average difference between TI and QPI : -2.241570
Total number of sentences taken into account : 106/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	2	0.607523	0.607523	0.474646	10.125375 %
4	9	0.441496	-0.156182	0.411790	7.358261 %
5	12	0.988103	-0.988103	0.398337	16.468376 %
6	32	2.047383	-2.047383	0.290881	34.123052 %
7	51	3.138086	-3.138086	0.472306	52.301427 %
	106	2.288720	-2.241570	0.915144	38.145340 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 5.027778
Average TI for the document : 3.869538
Average difference between TI and QPI : -1.158240
Total number of sentences taken into account : 36/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	1.733952	1.733952	0.471798	28.899204 %
3	7	0.662653	0.604538	0.551742	11.044219 %
4	2	0.394854	0.394854	0.053415	6.580908 %
5	6	1.211525	-1.211525	0.219025	20.192092 %
6	5	2.036353	-2.036353	0.402977	33.939213 %
7	12	3.016917	-3.016917	0.152846	50.281946 %
	36	1.833834	-1.158240	1.528423	30.563893 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.500000
Average TI for the document : 3.762016
Average difference between TI and QPI : -0.737984
Total number of sentences taken into account : 18/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	1.728531	1.728531	0.000000	28.808850 %
2	2	1.573601	1.573601	0.143180	26.226675 %
3	5	0.743423	0.731865	0.484613	12.390380 %
4	1	0.114392	0.114392	0.000000	1.906533 %
5	0	NaN	NaN	NaN	NaN %
6	6	2.175320	-2.175320	0.349117	36.255336 %
7	3	2.960414	-2.960414	0.311447	49.340233 %
	18	1.702245	-0.737984	1.699034	28.370742 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.083601
Average TI for the document : 3.958030
Average difference between TI and QPI : -1.125571
Total number of sentences taken into account : 311/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	3.011605	3.011605	0.000000	50.193417 %
2	20	1.732001	1.732001	0.295645	28.866688 %
3	31	0.707126	0.707126	0.301130	11.785431 %
4	58	0.274068	-0.074326	0.269401	4.567808 %
5	65	0.851950	-0.850755	0.276313	14.199174 %
6	62	1.965406	-1.965406	0.266103	32.756759 %
7	74	3.083244	-3.083244	0.379740	51.387406 %
	311	1.546175	-1.125571	1.285051	25.769588 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.560311
Average TI for the document : 3.853993
Average difference between TI and QPI : -1.706318
Total number of sentences taken into account : 257/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	3.160772	3.160772	0.000000	52.679533 %
2	8	1.495116	1.495116	0.478778	24.918600 %
3	15	0.747791	0.747791	0.305217	12.463183 %
4	24	0.334769	-0.101562	0.328156	5.579490 %
5	43	1.145222	-1.145222	0.331512	19.087037 %
6	106	1.964337	-1.964337	0.333650	32.738950 %
7	60	3.416008	-3.416008	0.709528	56.933467 %
	257	1.933065	-1.706318	1.062954	32.217756 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.125000
Average TI for the document : 3.943179
Average difference between TI and QPI : -0.181821
Total number of sentences taken into account : 32/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	1.647916	1.647916	0.365337	27.465273 %
3	9	0.827830	0.827830	0.371863	13.797169 %
4	1	0.091845	0.091845	0.000000	1.530750 %
5	13	0.974002	-0.974002	0.307646	16.233374 %
6	2	1.931017	-1.931017	0.037108	32.183617 %
7	2	2.538049	-2.538049	0.388183	42.300817 %
	32	1.168189	-0.181821	1.156826	19.469824 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

The calculations are made with all sentences of the training set but use only 5 factors.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258

Test set :

amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32
clin2test.stattix	sentence number :	176
dage7test.stattix	sentence number :	414
papert.stattix	sentence number :	78

File 1 : engnote1.stattix sentence number : 339
File 2 : microsof.stattix sentence number : 119
File 3 : nordson.stattix sentence number : 258

RESULTS OF THE WEIGHTS CALCULATION
==========

constant 1.983033

Fully Common :

1	Number of the Sentence	factor excluded by the user
2	Qpi	factor excluded by the user
3	Length	factor excluded by the user
4	Broken Sentences	factor excluded by the user
5	Short Parens	factor excluded by the user
6	Long Parens	factor excluded by the user
7	Coord. Conj	factor excluded by the user
8	Homographs	0.096279
9	No Verb	factor excluded by the user
10	All Upper Case	factor excluded by the user
11	Source language	factor excluded by the user
12	Marked Interrog	factor excluded by the user
13	WH Interrog	factor excluded by the user
14	Suspicious EOS	factor excluded by the user
15	SWORK to Verb	factor excluded by the user
16	Dble-quoted Verb SWORK	factor excluded by the user
17	Unmatched Left Parenth	factor excluded by the user
18	Not WC20 Count	0.012158
19	All Initial Caps	factor excluded by the user

Mixed :

1	Dependent Clause	factor excluded by the user
2	DC Secondary	factor excluded by the user
3	Nested Parens in REL	factor excluded by the user
4	REL Clauses	factor excluded by the user
5	Commas	factor excluded by the user
6	Tagged Interrog	factor excluded by the user
7	TH Clauses	factor excluded by the user
8	Unfound Words	0.122193

Source Specific :

1	No Main Clause	factor excluded by the user
2	Compound Clause	factor excluded by the user
3	INGs in Sent.	0.382661
4	ING Clause	factor excluded by the user
5	AS	factor excluded by the user
6	BOS AS	factor excluded by the user
7	WITH	factor excluded by the user
8	Long NP	factor excluded by the user
9	ING NP	factor excluded by the user

```
10          Quotes/Bold/Etc.    factor excluded by the user
11        Initial CapStrings    factor excluded by the user
12    All Upper Case_Strings    factor excluded by the user
13             Does not exist   factor excluded by the user
14             Does not exist   factor excluded by the user
15             Does not exist   factor excluded by the user
16             Does not exist   factor excluded by the user
```

Number of factors taken into account : 4/41
Total number of sentences taken into account : 637/716

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	2	4.836073	-4.836073	0.168735	80.601219 %
2	49	2.683934	-2.683934	0.532724	44.732238 %
3	68	1.594151	-1.594151	0.549018	26.569189 %
4	96	0.835036	-0.759491	0.439904	13.917262 %
5	154	0.419925	0.046306	0.423339	6.998750 %
6	152	0.906419	0.906419	0.416426	15.106976 %
7	116	1.530962	1.530962	0.291587	25.516039 %
	637	1.114264	0.000000	1.114264	18.571071 %

N.B. : all differences are calculated by doing TI-QPI

Significant factors

```
CO  0                  Constant 0.000000
FC  8              Homographs 0.000303
MX  8            Unfound Words 0.246569
```

Ideal set of factors :

```
CO  0                  Constant
FC  8              Homographs
MX  8            Unfound Words
FC 18           Not WC20 Count
SS  3            INGs in Sent.
```

Filename : amdahl.stattix

Average QPI for the document : 5.758621
Average TI for the document : 5.216336
Average difference between TI and QPI : -0.542285
Total number of sentences taken into account : 116/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	3.087696	3.087696	0.000000	51.461600 %
3	6	2.161196	2.161196	0.329242	36.019933 %
4	9	0.762586	0.669531	0.476967	12.709767 %
5	26	0.560071	-0.000997	0.560224	9.334512 %
6	36	0.830924	-0.830924	0.341672	13.848727 %
7	38	1.448592	-1.448592	0.325820	24.143202 %
	116	1.055515	-0.542285	0.858246	17.591916 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.141509
Average TI for the document : 5.195276
Average difference between TI and QPI : -0.946233
Total number of sentences taken into account : 106/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	2	2.015427	2.015427	0.314445	33.590442 %
4	9	0.779809	0.724248	0.399849	12.996809 %
5	12	0.414863	-0.146564	0.395955	6.914382 %
6	32	1.074870	-1.074870	0.428003	17.914497 %
7	51	1.464611	-1.464611	0.352005	24.410189 %
	106	1.180363	-0.946233	0.666937	19.672720 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 5.027778
Average TI for the document : 5.095450
Average difference between TI and QPI : 0.067672
Total number of sentences taken into account : 36/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	3.000197	3.000197	0.539178	50.003283 %
3	7	1.437492	1.264548	0.808847	23.958193 %
4	2	1.266192	1.266192	0.169534	21.103192 %
5	6	0.299171	-0.177258	0.327810	4.986178 %
6	5	0.782744	-0.782744	0.404326	13.045740 %
7	12	1.330962	-1.330962	0.193712	22.182696 %
	36	1.285442	0.067672	1.300480	21.424027 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.500000
Average TI for the document : 4.732059
Average difference between TI and QPI : 0.232059
Total number of sentences taken into account : 18/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.004809	5.004809	0.000000	83.413483 %
2	2	2.624781	2.624781	0.354479	43.746342 %
3	5	1.221083	1.221083	0.601039	20.351380 %
4	1	0.355147	-0.355147	0.000000	5.919117 %
5	0	NaN	NaN	NaN	NaN %
6	6	1.107665	-1.107665	0.782782	18.461086 %
7	3	1.727195	-1.727195	0.364596	28.786578 %
	18	1.585695	0.232059	1.635721	26.428246 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.083601
Average TI for the document : 5.060456
Average difference between TI and QPI : -0.023145
Total number of sentences taken into account : 311/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.667340	4.667340	0.000000	77.789000 %
2	20	2.843591	2.843591	0.665990	47.393185 %
3	31	1.669117	1.669117	0.562111	27.818622 %
4	58	0.840405	0.717407	0.475906	14.006750 %
5	65	0.418848	0.039002	0.414647	6.980793 %
6	62	0.878735	-0.878735	0.416491	14.645589 %
7	74	1.488421	-1.488421	0.318314	24.807014 %
	311	1.137862	-0.023145	1.135406	18.964368 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.560311
Average TI for the document : 5.128170
Average difference between TI and QPI : -0.432142
Total number of sentences taken into account : 257/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.679498	4.679498	0.000000	77.991633 %
2	8	2.699265	2.699265	0.730017	44.987746 %
3	15	1.496757	1.496757	0.428755	24.945944 %
4	24	0.837167	0.768910	0.467513	13.952785 %
5	43	0.425239	-0.156294	0.422951	7.087309 %
6	106	0.846038	-0.846038	0.340390	14.100627 %
7	60	1.363976	-1.363976	0.271543	22.732930 %
	257	1.006306	-0.432142	0.834640	16.771775 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.125000
Average TI for the document : 4.839714
Average difference between TI and QPI : 0.714714
Total number of sentences taken into account : 32/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.704298	2.704298	0.722196	45.071633 %
3	9	1.765550	1.765550	0.501371	29.425833 %
4	1	1.003575	1.003575	0.000000	16.726250 %
5	13	0.635053	-0.190770	0.637886	10.584214 %
6	2	0.833277	-0.833277	0.235111	13.887950 %
7	2	1.698804	-1.698804	0.257707	28.313400 %
	32	1.366715	0.714714	1.230894	22.778575 %

rem : all differences are calculated by doing TI-QPI

Filename : clin2test.stattix

Average QPI for the document : 5.222222
Average TI for the document : 4.867308
Average difference between TI and QPI : -0.354914
Total number of sentences taken into account : 162/176

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	2	1.792589	1.792589	0.089401	29.876483 %
3	18	1.679788	1.666647	0.487205	27.996464 %
4	23	0.515646	0.316906	0.401672	8.594107 %
5	41	0.574376	-0.243822	0.587235	9.572930 %
6	55	1.085672	-1.085672	0.328746	18.094540 %
7	23	1.246132	-1.246132	0.213227	20.768870 %
	162	0.972862	-0.354914	0.886522	16.214368 %

rem : all differences are calculated by doing TI-QPI

Filename : dage7test.stattix

Average QPI for the document : 5.580402
Average TI for the document : 4.953878
Average difference between TI and QPI : -0.626524
Total number of sentences taken into account : 398/414

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	3.192058	3.192058	0.847201	53.200963 %
3	23	1.592038	1.561359	0.438591	26.533973 %
4	53	0.708820	0.650964	0.416144	11.813663 %
5	73	0.496158	-0.352702	0.433384	8.269306 %
6	143	0.991117	-0.991117	0.415514	16.518615 %
7	101	1.665863	-1.665863	0.370803	27.764388 %
	398	1.096347	-0.626524	0.880990	18.272449 %

rem : all differences are calculated by doing TI-QPI

Filename : papert.stattix

Average QPI for the document : 4.410959
Average TI for the document : 4.981574
Average difference between TI and QPI : 0.570616
Total number of sentences taken into account : 73/78

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	6	2.036871	2.036871	1.017995	33.947844 %
3	15	1.701121	1.701121	0.509376	28.352012 %
4	13	1.022819	1.022819	0.401369	17.046983 %
5	23	0.377731	0.094168	0.368786	6.295516 %
6	14	0.647078	-0.647078	0.300563	10.784635 %
7	2	1.243259	-1.243259	0.115315	20.720983 %
	73	0.976275	0.570616	0.895703	16.271257 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

The calculations are made with all sentences of the training set but use only 10 factors.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258

Test set :

amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32
clin2test.stattix	sentence number :	176
dage7test.stattix	sentence number :	414
papert.stattix	sentence number :	78

```
File   1 :     engnote1.stattix      sentence number :  339
File   2 :     microsof.stattix      sentence number :  119
File   3 :      nordson.stattix      sentence number :  258

RESULTS OF THE WEIGHTS CALCULATION
==================================

                    constant      2.022118

Fully Common :
------------
 1    Number of the Sentence   factor excluded by the user
 2                       Qpi   factor excluded by the user
 3                    Length   factor excluded by the user
 4          Broken Sentences   factor excluded by the user
 5              Short Parens   factor excluded by the user
 6               Long Parens   factor excluded by the user
 7               Coord. Conj   factor excluded by the user
 8                Homographs      0.100812
 9                   No Verb   factor excluded by the user
10            All Upper Case     -0.824371
11           Source language   factor excluded by the user
12           Marked Interrog      0.264482
13              WH Interrog   factor excluded by the user
14           Suspicious EOS   factor excluded by the user
15            SWORK to Verb   factor excluded by the user
16    Dble-quoted Verb SWORK      0.677346
17    Unmatched Left Parenth      1.106075
18             Not WC20 Count      0.006575
19          All Initial Caps   factor excluded by the user

Mixed :
-----
 1          Dependent Clause   factor excluded by the user
 2              DC Secondary   factor excluded by the user
 3      Nested Parens in REL   factor excluded by the user
 4               REL Clauses   factor excluded by the user
 5                    Commas   factor excluded by the user
 6           Tagged Interrog      1.596441
 7                TH Clauses   factor excluded by the user
 8             Unfound Words      0.145425

Source Specific :
---------------
 1            No Main Clause   factor excluded by the user
 2           Compound Clause   factor excluded by the user
 3             INGs in Sent.      0.380314
 4                ING Clause   factor excluded by the user
 5                        AS   factor excluded by the user
 6                    BOS AS     -0.137500
 7                      WITH   factor excluded by the user
 8                   Long NP   factor excluded by the user
 9                    ING NP   factor excluded by the user
```

```
10          Quotes/Bold/Etc.   factor excluded by the user
11        Initial CapStrings   factor excluded by the user
12    All Upper Case_Strings   factor excluded by the user
13            Does not exist   factor excluded by the user
14            Does not exist   factor excluded by the user
15            Does not exist   factor excluded by the user
16            Does not exist   factor excluded by the user


Number of factors taken into account          : 10/41
Total number of sentences taken into account : 637/716
```

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	2	4.806939	4.806939	9.613879	80.115654 %
2	49	2.667489	2.667489	5.334977	44.458143 %
3	68	1.561379	1.532482	3.064964	26.022988 %
4	96	0.823281	0.760602	1.535949	13.721356 %
5	154	0.416696	-0.051934	0.414831	6.944933 %
6	152	0.903504	-0.897360	1.794720	15.058401 %
7	116	1.492679	-1.492679	2.985358	24.877982 %
	637	1.099190	-0.000000	1.099190	18.319837 %

N.B. : all differences are calculated by doing TI-QPI

Significant factors

```
CO  0                  Constant 0.000000
FC  8               Homographs 0.000164
SS  3            INGs in Sent. 0.007914
FC 10           All Upper Case 0.035665
MX  6         Tagged Interrog 0.108050
MX  8           Unfound Words 0.169344
FC 16  Dble-quoted Verb SWORK 0.235280
FC 17  Unmatched Left Parenth 0.259587
FC 12         Marked Interrog 0.284220
```

Ideal set of factors :

```
CO  0                  Constant
FC  8               Homographs
SS  3            INGs in Sent.
```

Filename : amdahl.stattix

Average QPI for the document : 5.765217
Average TI for the document : 5.221448
Average difference between TI and QPI : -0.543770
Total number of sentences taken into account : 115/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	3.134123	3.134123	0.000000	52.235383 %
3	6	2.175974	2.175974	0.305881	36.266231 %
4	9	0.769843	0.705742	0.466205	12.830713 %
5	25	0.526797	-0.004536	0.527342	8.779957 %
6	36	0.818554	-0.818554	0.326347	13.642569 %
7	38	1.460363	-1.460363	0.311692	24.339389 %
	115	1.054350	-0.543770	0.855878	17.572499 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.133333
Average TI for the document : 5.314891
Average difference between TI and QPI : -0.818443
Total number of sentences taken into account : 105/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	2	2.006624	2.006624	0.307211	33.443725 %
4	9	0.794547	0.755826	0.382272	13.242454 %
5	12	0.392284	-0.117437	0.383225	6.538065 %
6	32	1.056321	-1.056321	0.415219	17.605356 %
7	50	1.230813	-1.230813	0.583558	20.513549 %
	105	1.059186	-0.818443	0.744914	17.653097 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 5.000000
Average TI for the document : 5.057168
Average difference between TI and QPI : 0.057168
Total number of sentences taken into account : 35/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	3.008326	3.008326	0.525370	50.138758 %
3	7	1.314507	1.169754	0.733367	21.908455 %
4	2	1.125631	1.125631	0.312340	18.760508 %
5	6	0.299524	-0.179947	0.315448	4.992075 %
6	4	0.893308	-0.893308	0.392330	14.888475 %
7	12	1.318252	-1.318252	0.258869	21.970871 %
	35	1.276443	0.057168	1.287877	21.274058 %

rem : all differences are calculated by doing TI-QPI

Filename : clin2test.stattix

Average QPI for the document : 5.222222
Average TI for the document : 4.955730
Average difference between TI and QPI : -0.266492
Total number of sentences taken into account : 162/176

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	2	1.824506	1.824506	0.081793	30.408433 %
3	18	1.640328	1.563212	0.604104	27.338795 %
4	23	0.507632	0.325834	0.383887	8.460526 %
5	41	0.583773	-0.228370	0.588257	9.729552 %
6	55	1.091846	-1.091846	0.326030	18.197432 %
7	23	0.874913	-0.566874	0.799876	14.581889 %
	162	0.919504	-0.266492	0.869910	15.325069 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.500000
Average TI for the document : 4.773455
Average difference between TI and QPI : 0.273455
Total number of sentences taken into account : 18/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.971307	4.971307	0.000000	82.855117 %
2	2	2.699604	2.699604	0.327131	44.993408 %
3	5	1.289885	1.289885	0.574266	21.498077 %
4	1	0.267228	-0.267228	0.000000	4.453800 %
5	0	NaN	NaN	NaN	NaN %
6	6	1.088031	-1.088031	0.751056	18.133853 %
7	3	1.700778	-1.700778	0.337990	28.346294 %
	18	1.595427	0.273455	1.642401	26.590451 %

rem : all differences are calculated by doing TI-QPI

Filename : consumer.stattix

Average QPI for the document : 4.643836
Average TI for the document : 4.919445
Average difference between TI and QPI : 0.275610
Total number of sentences taken into account : 73/76

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	9	2.843367	2.843367	0.597724	47.389443 %
3	8	1.761868	1.761868	0.361979	29.364467 %
4	13	0.929648	0.846196	0.437524	15.494126 %
5	18	0.391204	-0.129255	0.405041	6.520069 %
6	20	1.055313	-1.055313	0.456407	17.588554 %
7	5	1.426684	-1.426684	0.219333	23.778063 %
	73	1.192493	0.275610	1.204403	19.874890 %

rem : all differences are calculated by doing TI-QPI

Filename : dage7test.stattix

Average QPI for the document : 5.580402
Average TI for the document : 5.009010
Average difference between TI and QPI : -0.571392
Total number of sentences taken into account : 398/414

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.843626	2.843626	1.100748	47.393770 %
3	23	1.479624	1.456701	0.521115	24.660399 %
4	53	0.671558	0.613184	0.393158	11.192637 %
5	73	0.460783	-0.310579	0.415362	7.679709 %
6	143	1.007775	-0.926155	0.440646	16.796253 %
7	101	1.573697	-1.510126	0.517518	26.228282 %
	398	1.056619	-0.571392	0.885283	17.610320 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.080645
Average TI for the document : 5.047954
Average difference between TI and QPI : -0.032691
Total number of sentences taken into account : 310/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.642571	4.642571	0.000000	77.376183 %
2	20	2.775851	2.775851	0.761331	46.264182 %
3	31	1.626283	1.569989	0.644605	27.104719 %
4	58	0.804064	0.700458	0.443572	13.401066 %
5	65	0.408704	-0.002869	0.408924	6.811728 %
6	61	0.883910	-0.868600	0.401665	14.731827 %
7	74	1.438093	-1.438093	0.349826	23.968215 %
	310	1.110043	-0.032691	1.106839	18.500708 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.554688
Average TI for the document : 5.138284
Average difference between TI and QPI : -0.416404
Total number of sentences taken into account : 256/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.649146	4.649146	0.000000	77.485767 %
2	8	2.717522	2.717522	0.703215	45.292037 %
3	15	1.554843	1.554843	0.400079	25.914050 %
4	24	0.839972	0.800963	0.439927	13.999537 %
5	43	0.405888	-0.130337	0.405836	6.764805 %
6	106	0.829517	-0.829517	0.324606	13.825285 %
7	59	1.389851	-1.389851	0.259837	23.164178 %
	256	1.004900	-0.416404	0.842524	16.748340 %

rem : all differences are calculated by doing TI-QPI

Filename : microsof.stattix

Average QPI for the document : 5.522124
Average TI for the document : 5.044462
Average difference between TI and QPI : -0.477662
Total number of sentences taken into account : 113/119

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	2.685871	2.685871	0.117445	44.764517 %
3	11	1.499322	1.479330	0.646582	24.988695 %
4	8	0.575982	0.574961	0.237591	9.599696 %
5	20	0.420264	-0.047877	0.434627	7.004406 %
6	39	0.923335	-0.923335	0.414049	15.388918 %
7	31	1.568517	-1.568517	0.225437	26.141952 %
	113	1.105162	-0.477662	0.968484	18.419360 %

rem : all differences are calculated by doing TI-QPI

Filename : nordson.stattix

Average QPI for the document : 4.898833
Average TI for the document : 5.048600
Average difference between TI and QPI : 0.149767
Total number of sentences taken into account : 257/258

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.971307	4.971307	0.000000	82.855117 %
2	27	2.668167	2.668167	0.504930	44.469451 %
3	27	1.552670	1.552670	0.539989	25.877831 %
4	30	0.926378	0.926378	0.369527	15.439636 %
5	72	0.453857	-0.046897	0.456998	7.564286 %
6	57	0.856778	-0.856778	0.393254	14.279627 %
7	43	1.302842	-1.302842	0.253597	21.714033 %
	257	1.106076	0.149767	1.120937	18.434598 %

rem : all differences are calculated by doing TI-QPI

Filename : papert.stattix

Average QPI for the document : 4.410959
Average TI for the document : 4.963227
Average difference between TI and QPI : 0.552268
Total number of sentences taken into account : 73/78

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	6	1.848719	1.848719	0.988972	30.811986 %
3	15	1.748792	1.748792	0.499068	29.146537 %
4	13	0.980253	0.980253	0.380640	16.337542 %
5	23	0.387397	0.071406	0.375069	6.456624 %
6	14	0.631641	-0.631641	0.287317	10.527348 %
7	2	1.275636	-1.275636	0.126406	21.260600 %
	73	0.963998	0.552268	0.892592	16.066636 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.032258
Average TI for the document : 4.856612
Average difference between TI and QPI : 0.824354
Total number of sentences taken into account : 31/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.745182	2.745182	0.692588	45.753037 %
3	9	1.814076	1.814076	0.464005	30.234600 %
4	1	1.039886	1.039886	0.000000	17.331433 %
5	13	0.599651	-0.148119	0.606729	9.994176 %
6	2	0.823139	-0.823139	0.224636	13.718992 %
7	1	1.965667	-1.965667	0.000000	32.761117 %
	31	1.370964	0.824354	1.208204	22.849400 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

For all files, the training set is constituted with all of the other files.

the distribution of the qpi is natural and all factors are used.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258
amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32
clin2test.stattix	sentence number :	176
dage7test.stattix	sentence number :	414
papert.stattix	sentence number :	78

Filename : amdahl.stattix

Average QPI for the document : 5.758621
Average TI for the document : 5.390619
Average difference between TI and QPI : -0.368002
Total number of sentences taken into account : 116/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	3.455427	3.455427	0.000000	57.590450 %
3	6	2.400211	2.400211	0.407993	40.003511 %
4	9	0.914857	0.914857	0.470309	15.247617 %
5	26	0.539490	0.205320	0.468476	8.991493 %
6	36	0.657545	-0.641299	0.323211	10.959087 %
7	38	1.342899	-1.342899	0.309279	22.381649 %
	116	0.989818	-0.368002	0.879017	16.496971 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.141509
Average TI for the document : 5.477824
Average difference between TI and QPI : -0.663686
Total number of sentences taken into account : 106/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	2	2.245275	2.245275	0.327901	37.421250 %
4	9	1.024675	0.998256	0.359863	17.077915 %
5	12	0.324578	0.184213	0.306393	5.409632 %
6	32	0.850109	-0.850109	0.370072	14.168480 %
7	51	1.153581	-1.153581	0.453673	19.226351 %
	106	0.977771	-0.663686	0.664057	16.296175 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 5.027778
Average TI for the document : 5.187747
Average difference between TI and QPI : 0.159969
Total number of sentences taken into account : 36/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	3.142532	3.142532	0.543835	52.375525 %
3	7	1.167237	1.167237	0.674023	19.453943 %
4	2	1.171102	1.171102	0.520073	19.518367 %
5	6	0.287639	0.045215	0.276322	4.793981 %
6	5	0.627184	-0.627184	0.415976	10.453073 %
7	12	1.204956	-1.204956	0.177552	20.082608 %
	36	1.177895	0.159969	1.193783	19.631583 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.500000
Average TI for the document : 5.128084
Average difference between TI and QPI : 0.628084
Total number of sentences taken into account : 18/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.576049	5.576049	0.000000	92.934150 %
2	2	2.804093	2.804093	0.382160	46.734883 %
3	5	1.646086	1.646086	0.457734	27.434767 %
4	1	0.469359	0.469359	0.000000	7.822650 %
5	0	NaN	NaN	NaN	NaN %
6	6	0.759566	-0.716608	0.573716	12.659433 %
7	3	1.426290	-1.426290	0.245134	23.771494 %
	18	1.595572	0.628084	1.598888	26.592860 %

rem : all differences are calculated by doing TI-QPI

Filename : consumer.stattix

Average QPI for the document : 4.675676
Average TI for the document : 5.172846
Average difference between TI and QPI : 0.497170
Total number of sentences taken into account : 74/76

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	9	3.094803	3.094803	0.421917	51.580044 %
3	8	1.989464	1.989464	0.265472	33.157727 %
4	13	1.060759	1.053732	0.417479	17.679310 %
5	18	0.438084	0.111096	0.401053	7.301407 %
6	20	0.828183	-0.819355	0.430673	13.803053 %
7	6	1.048244	-1.048244	0.262830	17.470725 %
	74	1.193209	0.497170	1.186684	19.886811 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.125000
Average TI for the document : 5.102570
Average difference between TI and QPI : 0.977570
Total number of sentences taken into account : 32/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	3.009250	3.009250	0.777750	50.154167 %
3	9	2.036367	2.036367	0.554466	33.939452 %
4	1	1.261476	1.261476	0.000000	21.024600 %
5	13	0.610205	0.082917	0.591070	10.170077 %
6	2	0.641518	-0.641518	0.186238	10.691967 %
7	2	1.573842	-1.573842	0.762942	26.230708 %
	32	1.468700	0.977570	1.248218	24.478339 %

rem : all differences are calculated by doing TI-QPI

Filename : microsof.stattix

Average QPI for the document : 5.535088
Average TI for the document : 5.294401
Average difference between TI and QPI : -0.240687
Total number of sentences taken into account : 114/119

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	2.956656	2.956656	0.112119	49.277596 %
3	11	1.831492	1.759986	0.565392	30.524859 %
4	8	0.905840	0.905840	0.252945	15.097335 %
5	20	0.453251	0.231731	0.378165	7.554177 %
6	39	0.725335	-0.695418	0.388598	12.088919 %
7	32	1.355775	-1.355775	0.219246	22.596252 %
	114	1.052260	-0.240687	0.982475	17.537666 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.083601
Average TI for the document : 5.154995
Average difference between TI and QPI : 0.071394
Total number of sentences taken into account : 311/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.830485	4.830485	0.000000	80.508083 %
2	20	2.831733	2.831733	1.035214	47.195542 %
3	31	1.968297	1.728264	0.838688	32.804946 %
4	58	1.067467	0.893501	0.596495	17.791123 %
5	65	0.664796	0.044571	0.645174	11.079937 %
6	62	0.822230	-0.811672	0.608304	13.703826 %
7	74	1.313979	-1.313979	0.458025	21.899647 %
	311	1.208424	0.071394	1.211758	20.140400 %

rem : all differences are calculated by doing TI-QPI

Filename : nordson.stattix

Average QPI for the document : 4.906977
Average TI for the document : 5.314450
Average difference between TI and QPI : 0.407473
Total number of sentences taken into account : 258/258

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.491524	5.491524	0.000000	91.525400 %
2	27	2.997827	2.997827	0.496826	49.963786 %
3	27	1.826386	1.826386	0.552829	30.439766 %
4	30	1.113968	1.078147	0.433775	18.566128 %
5	72	0.484457	0.192931	0.437575	8.074281 %
6	57	0.654257	-0.629523	0.380473	10.904281 %
7	44	0.931134	-0.931134	0.303550	15.518906 %
	258	1.094216	0.407473	1.113523	18.236935 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.560311
Average TI for the document : 5.318919
Average difference between TI and QPI : -0.241392
Total number of sentences taken into account : 257/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.859363	4.859363	0.000000	80.989383 %
2	8	2.997141	2.997141	0.619776	49.952342 %
3	15	1.675640	1.674002	0.514878	27.927340 %
4	24	1.012956	0.937096	0.526675	16.882595 %
5	43	0.333601	0.053946	0.324257	5.560012 %
6	106	0.731369	-0.710043	0.331798	12.189489 %
7	60	1.092162	-1.092162	0.423407	18.202705 %
	257	0.917050	-0.241392	0.831748	15.284160 %

rem : all differences are calculated by doing TI-QPI

Filename : clin2test.stattix

Average QPI for the document : 5.222222
Average TI for the document : 4.520953
Average difference between TI and QPI : -0.701269
Total number of sentences taken into account : 162/176

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	2	1.666220	1.666220	0.238821	27.770333 %
3	18	1.390061	1.021568	0.949160	23.167688 %
4	23	0.698512	-0.150101	0.675755	11.641874 %
5	41	0.918843	-0.724423	0.756255	15.314055 %
6	55	1.455340	-1.455340	0.601720	24.255674 %
7	23	0.962758	-0.962125	0.658575	16.045963 %
	162	1.137525	-0.701269	0.918636	18.958751 %

rem : all differences are calculated by doing TI-QPI

Filename : dage7test.stattix

Average QPI for the document : 5.583960
Average TI for the document : 4.999275
Average difference between TI and QPI : -0.584685
Total number of sentences taken into account : 399/414

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	3.000431	2.534775	1.592637	50.007183 %
3	23	1.574503	1.569996	0.652066	26.241725 %
4	53	0.712285	0.679111	0.414995	11.871416 %
5	73	0.421531	-0.212792	0.416520	7.025513 %
6	143	0.944862	-0.944862	0.396228	15.747696 %
7	102	1.671368	-1.641342	0.547643	27.856139 %
	399	1.065998	-0.584685	0.903143	17.766641 %

rem : all differences are calculated by doing TI-QPI

Filename : papert.stattix

Average QPI for the document : 4.445946
Average TI for the document : 5.202537
Average difference between TI and QPI : 0.756591
Total number of sentences taken into account : 74/78

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	6	1.859773	1.847993	0.778588	30.996211 %
3	15	1.992281	1.992281	0.448724	33.204683 %
4	13	1.296847	1.296847	0.381005	21.614121 %
5	23	0.497513	0.316046	0.362796	8.291882 %
6	14	0.434127	-0.426374	0.289343	7.235456 %
7	3	1.047757	-1.047757	0.110425	17.462622 %
	74	1.061699	0.756591	0.920099	17.694981 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

For training and testing sets, no sentences with <=3 words were included.
Otherwise, for each test document, train with all other docs.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258
amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32
clin2test.stattix	sentence number :	176
dage7test.stattix	sentence number :	414
papert.stattix	sentence number :	78

Filename : amdahl.stattix

Average QPI for the document : 5.673913
Average TI for the document : 5.358734
Average difference between TI and QPI : -0.315179
Total number of sentences taken into account : 92/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	3.461617	3.461617	0.000000	57.693617 %
3	6	2.414556	2.414556	0.402516	40.242597 %
4	7	0.861876	0.861876	0.557472	14.364595 %
5	21	0.518278	0.192145	0.444495	8.637970 %
6	30	0.607518	-0.607518	0.281701	10.125301 %
7	27	1.436595	-1.436595	0.279777	23.943247 %
	92	0.998690	-0.315179	0.908822	16.644841 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.084337
Average TI for the document : 5.461053
Average difference between TI and QPI : -0.623284
Total number of sentences taken into account : 83/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	1	1.913021	1.913021	0.000000	31.883683 %
4	8	1.018262	0.988193	0.368903	16.971035 %
5	11	0.332621	0.230906	0.306963	5.543679 %
6	26	0.814717	-0.814717	0.329677	13.578611 %
7	37	1.159689	-1.159689	0.496613	19.328145 %
	83	0.937459	-0.623284	0.685767	15.624309 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 4.827586
Average TI for the document : 5.006496
Average difference between TI and QPI : 0.178910
Total number of sentences taken into account : 29/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	3.050966	3.050966	0.451519	50.849438 %
3	6	0.964582	0.964582	0.580106	16.076361 %
4	1	1.631966	1.631966	0.000000	27.199433 %
5	6	0.277931	0.031048	0.267582	4.632186 %
6	4	0.844757	-0.844757	0.296248	14.079279 %
7	8	1.405274	-1.405274	0.174967	23.421225 %
	29	1.238349	0.178910	1.241778	20.639151 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.666667
Average TI for the document : 4.914178
Average difference between TI and QPI : 0.247512
Total number of sentences taken into account : 15/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	2	2.658876	2.658876	0.484893	44.314592 %
3	4	1.402956	1.402956	0.600273	23.382600 %
4	1	0.267048	0.267048	0.000000	4.450800 %
5	0	NaN	NaN	NaN	NaN %
6	6	0.832261	-0.813149	0.631211	13.871025 %
7	2	1.302525	-1.302525	0.215601	21.708742 %
	15	1.253016	0.247512	1.261872	20.883601 %

rem : all differences are calculated by doing TI-QPI

Filename : consumer.stattix

Average QPI for the document : 4.638889
Average TI for the document : 5.128363
Average difference between TI and QPI : 0.489474
Total number of sentences taken into account : 72/76

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	9	3.046001	3.046001	0.454241	50.766687 %
3	8	1.967634	1.967634	0.255075	32.793894 %
4	13	1.049359	1.033983	0.427493	17.489315 %
5	17	0.434862	0.082190	0.410689	7.247706 %
6	20	0.836795	-0.827994	0.434499	13.946580 %
7	5	1.238418	-1.238418	0.125993	20.640297 %
	72	1.209964	0.489474	1.204701	20.166063 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.032967
Average TI for the document : 5.117533
Average difference between TI and QPI : 0.084566
Total number of sentences taken into account : 273/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.840781	4.840781	0.000000	80.679683 %
2	17	2.844939	2.844939	0.945338	47.415657 %
3	26	2.051069	1.772571	0.816329	34.184476 %
4	56	1.053942	0.876679	0.584080	17.565693 %
5	61	0.670156	0.043431	0.649332	11.169266 %
6	52	0.871775	-0.854002	0.649125	14.529576 %
7	60	1.392339	-1.392339	0.471305	23.205653 %
	273	1.228225	0.084566	1.228430	20.470415 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.538462
Average TI for the document : 5.239233
Average difference between TI and QPI : -0.299228
Total number of sentences taken into account : 221/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.867331	4.867331	0.000000	81.122183 %
2	6	3.151154	3.151154	0.429285	52.519236 %
3	13	1.608346	1.605172	0.538083	26.805765 %
4	22	1.027515	0.939856	0.547465	17.125249 %
5	39	0.341916	0.034494	0.336413	5.698596 %
6	91	0.761701	-0.760933	0.338707	12.695013 %
7	49	1.296901	-1.296901	0.327312	21.615017 %
	221	0.965999	-0.299228	0.866506	16.099979 %

rem : all differences are calculated by doing TI-QPI

Filename : microsof.stattix

Average QPI for the document : 5.518868
Average TI for the document : 5.288694
Average difference between TI and QPI : -0.230174
Total number of sentences taken into account : 106/119

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	2.966496	2.966496	0.105652	49.441604 %
3	10	1.860842	1.771610	0.615375	31.014038 %
4	8	0.897834	0.897834	0.244475	14.963906 %
5	18	0.474654	0.250376	0.383601	7.910903 %
6	37	0.736824	-0.734781	0.368369	12.280400 %
7	29	1.327004	-1.327004	0.189542	22.116740 %
	106	1.056099	-0.230174	0.989959	17.601647 %

rem : all differences are calculated by doing TI-QPI

Filename : nordson.stattix

Average QPI for the document : 4.893805
Average TI for the document : 5.295337
Average difference between TI and QPI : 0.401531
Total number of sentences taken into account : 226/258

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.888813	4.888813	0.000000	81.480217 %
2	24	3.100526	3.100526	0.383618	51.675441 %
3	24	1.937589	1.937589	0.541075	32.293148 %
4	25	1.152346	1.070436	0.485276	19.205771 %
5	66	0.503666	0.241061	0.437690	8.394426 %
6	47	0.630856	-0.628672	0.356714	10.514271 %
7	39	1.235407	-1.235407	0.210490	20.590116 %
	226	1.175599	0.401531	1.183657	19.593310 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.066667
Average TI for the document : 4.999442
Average difference between TI and QPI : 0.932776
Total number of sentences taken into account : 30/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.870873	2.870873	0.671608	47.847877 %
3	8	1.977863	1.977863	0.462903	32.964383 %
4	1	1.199259	1.199259	0.000000	19.987650 %
5	13	0.597809	0.027098	0.591556	9.963488 %
6	2	0.690505	-0.690505	0.191626	11.508425 %
7	1	2.364517	-2.364517	0.000000	39.408617 %
	30	1.429786	0.932776	1.221178	23.829764 %

rem : all differences are calculated by doing TI-QPI

Filename : clin2test.stattix

Average QPI for the document : 5.217687
Average TI for the document : 4.506939
Average difference between TI and QPI : -0.710748
Total number of sentences taken into account : 147/176

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	1.849191	1.849191	0.000000	30.819850 %
3	17	1.349806	0.924659	1.006067	22.496764 %
4	22	0.720517	-0.160595	0.689820	12.008616 %
5	36	0.917855	-0.736047	0.722726	15.297581 %
6	51	1.511782	-1.511782	0.635444	25.196363 %
7	20	0.920261	-0.745836	0.834886	15.337678 %
	147	1.150994	-0.710748	0.951799	19.183234 %

rem : all differences are calculated by doing TI-QPI

Filename : dage7test.stattix

Average QPI for the document : 5.552198
Average TI for the document : 4.980791
Average difference between TI and QPI : -0.571407
Total number of sentences taken into account : 364/414

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.965982	2.496426	1.567522	49.433040 %
3	23	1.565592	1.561128	0.646989	26.093199 %
4	48	0.722619	0.685195	0.425091	12.043644 %
5	65	0.435167	-0.238849	0.429418	7.252780 %
6	136	0.945683	-0.945683	0.406822	15.761384 %
7	87	1.722551	-1.668179	0.590218	28.709187 %
	364	1.077706	-0.571407	0.916443	17.961763 %

rem : all differences are calculated by doing TI-QPI

Filename : papert.stattix

Average QPI for the document : 4.442857
Average TI for the document : 5.177678
Average difference between TI and QPI : 0.734821
Total number of sentences taken into account : 70/78

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	1.712744	1.675198	0.867561	28.545730 %
3	15	1.966653	1.966653	0.459361	32.777550 %
4	12	1.301554	1.301554	0.416915	21.692571 %
5	22	0.475546	0.276222	0.363733	7.925764 %
6	14	0.434945	-0.425498	0.295420	7.249079 %
7	2	1.088463	-1.088463	0.110845	18.141042 %
	70	1.034433	0.734821	0.906593	17.240553 %

rem : all differences are calculated by doing TI-QPI

Type of calculation :

For training and testing sets, no sentences with qpi<3 were included.
Otherwise, for each test document, train with all other docs.

Training set :

engnote1.stattix	sentence number :	339
microsof.stattix	sentence number :	119
nordson.stattix	sentence number :	258
amdahl.stattix	sentence number :	125
berlitz.stattix	sentence number :	106
capoten1.stattix	sentence number :	36
comtec.stattix	sentence number :	21
consumer.stattix	sentence number :	76
infobull.stattix	sentence number :	269
schneider.stattix	sentence number :	32
clin2test.stattix	sentence number :	176
dage7test.stattix	sentence number :	414
papert.stattix	sentence number :	78

Filename : amdahl.stattix

Average QPI for the document : 5.765217
Average TI for the document : 5.589322
Average difference between TI and QPI : -0.175895
Total number of sentences taken into account : 115/125

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	1	3.482559	3.482559	0.000000	58.042650 %
3	6	2.507375	2.507375	0.368801	41.789581 %
4	9	1.115297	1.115297	0.379872	18.588278 %
5	25	0.565657	0.370521	0.429478	9.427625 %
6	36	0.552535	-0.472181	0.317551	9.208920 %
7	38	1.080445	-1.080445	0.397761	18.007416 %
	115	0.901340	-0.175895	0.845910	15.022332 %

rem : all differences are calculated by doing TI-QPI

Filename : berlitz.stattix

Average QPI for the document : 6.133333
Average TI for the document : 5.657066
Average difference between TI and QPI : -0.476268
Total number of sentences taken into account : 105/106

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	0	NaN	NaN	NaN	NaN %
3	2	2.423891	2.423891	0.280629	40.398183 %
4	9	1.171460	1.171460	0.311803	19.524328 %
5	12	0.381985	0.283694	0.290522	6.366422 %
6	32	0.686422	-0.686422	0.306390	11.440370 %
7	50	0.936757	-0.936757	0.571468	15.612615 %
	105	0.845506	-0.476268	0.672503	14.091762 %

rem : all differences are calculated by doing TI-QPI

Filename : capoten1.stattix

Average QPI for the document : 5.000000
Average TI for the document : 5.346450
Average difference between TI and QPI : 0.346450
Total number of sentences taken into account : 35/84

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	3.307743	3.307743	0.511726	55.129058 %
3	7	1.438500	1.438500	0.552031	23.974998 %
4	2	1.302711	1.302711	0.340248	21.711850 %
5	6	0.347835	0.232025	0.256319	5.797256 %
6	4	0.578318	-0.578318	0.321890	9.638633 %
7	12	1.071585	-1.071585	0.131762	17.859750 %
	35	1.233291	0.346450	1.243552	20.554857 %

rem : all differences are calculated by doing TI-QPI

Filename : comtec.stattix

Average QPI for the document : 4.500000
Average TI for the document : 5.180591
Average difference between TI and QPI : 0.680591
Total number of sentences taken into account : 18/21

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.866471	5.866471	0.000000	97.774517 %
2	2	2.793876	2.793876	0.510911	46.564608 %
3	5	1.762762	1.762762	0.443814	29.379360 %
4	1	0.381581	0.381581	0.000000	6.359683 %
5	0	NaN	NaN	NaN	NaN %
6	6	0.733469	-0.664503	0.570117	12.224481 %
7	3	1.470654	-1.470654	0.304824	24.510906 %
	18	1.636799	0.680591	1.647034	27.279990 %

rem : all differences are calculated by doing TI-QPI

Filename : consumer.stattix

Average QPI for the document : 4.643836
Average TI for the document : 5.267802
Average difference between TI and QPI : 0.623967
Total number of sentences taken into account : 73/76

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	9	3.211110	3.211110	0.436768	53.518502 %
3	8	2.058966	2.058966	0.284271	34.316106 %
4	13	1.159273	1.159273	0.396763	19.321222 %
5	18	0.386900	0.219132	0.303538	6.448332 %
6	20	0.680396	-0.663182	0.369463	11.339940 %
7	5	1.114688	-1.114688	0.164407	18.578130 %
	73	1.186135	0.623967	1.169262	19.768913 %

rem : all differences are calculated by doing TI-QPI

Filename : engnote1.stattix

Average QPI for the document : 5.080645
Average TI for the document : 5.319868
Average difference between TI and QPI : 0.239223
Total number of sentences taken into account : 310/339

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.063856	5.063856	0.000000	84.397600 %
2	20	3.027080	3.027080	0.985740	50.451339 %
3	31	2.125601	1.926270	0.831194	35.426684 %
4	58	1.151564	1.063291	0.557521	19.192739 %
5	65	0.652669	0.190436	0.573893	10.877813 %
6	61	0.693365	-0.664790	0.574724	11.556084 %
7	74	1.144026	-1.144026	0.450899	19.067093 %
	310	1.186021	0.239223	1.195110	19.767014 %

rem : all differences are calculated by doing TI-QPI

Filename : infobull.stattix

Average QPI for the document : 5.554688
Average TI for the document : 5.481176
Average difference between TI and QPI : -0.073511
Total number of sentences taken into account : 256/269

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	4.982602	4.982602	0.000000	83.043367 %
2	8	3.277586	3.277586	0.580481	54.626437 %
3	15	1.864387	1.864387	0.474617	31.073124 %
4	24	1.134820	1.107696	0.495904	18.913663 %
5	43	0.379320	0.238399	0.311986	6.321998 %
6	106	0.607504	-0.570944	0.314999	10.125070 %
7	59	0.920403	-0.920403	0.471757	15.340055 %
	256	0.864901	-0.073511	0.841123	14.415024 %

rem : all differences are calculated by doing TI-QPI

Filename : microsof.stattix

Average QPI for the document : 5.522124
Average TI for the document : 5.434504
Average difference between TI and QPI : -0.087620
Total number of sentences taken into account : 113/119

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	4	3.060271	3.060271	0.181516	51.004521 %
3	11	1.968187	1.968187	0.460823	32.803114 %
4	8	0.979449	0.979449	0.297270	16.324156 %
5	20	0.475274	0.349518	0.333377	7.921225 %
6	39	0.600780	-0.539302	0.356906	10.013003 %
7	31	1.212432	-1.212432	0.206349	20.207202 %
	113	0.993345	-0.087620	0.972935	16.555756 %

rem : all differences are calculated by doing TI-QPI

Filename : nordson.stattix

Average QPI for the document : 4.898833
Average TI for the document : 5.527064
Average difference between TI and QPI : 0.628231
Total number of sentences taken into account : 257/258

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	1	5.848105	5.848105	0.000000	97.468417 %
2	27	3.237524	3.237524	0.417010	53.958728 %
3	27	2.055951	2.055951	0.503847	34.265854 %
4	30	1.327143	1.240886	0.496344	22.119056 %
5	72	0.562729	0.395194	0.432395	9.378812 %
6	57	0.617541	-0.466587	0.455574	10.292353 %
7	43	0.617057	-0.613990	0.475789	10.284282 %
	257	1.131658	0.628231	1.127069	18.860959 %

rem : all differences are calculated by doing TI-QPI

Filename : schneider.stattix

Average QPI for the document : 4.032258
Average TI for the document : 5.153699
Average difference between TI and QPI : 1.121441
Total number of sentences taken into account : 31/32

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	3.157875	3.157875	0.721818	52.631257 %
3	9	2.118284	2.118284	0.488118	35.304741 %
4	1	1.294947	1.294947	0.000000	21.582450 %
5	13	0.583406	0.155552	0.547509	9.723426 %
6	2	0.512682	-0.512682	0.202437	8.544708 %
7	1	2.381037	-2.381037	0.000000	39.683950 %
	31	1.520631	1.121441	1.246921	25.343848 %

rem : all differences are calculated by doing TI-QPI

Filename : clin2test.stattix

Average QPI for the document : 5.222222
Average TI for the document : 4.957562
Average difference between TI and QPI : -0.264660
Total number of sentences taken into account : 162/176

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	2	2.205135	2.205135	0.175701	36.752250 %
3	18	1.678581	1.544589	0.708875	27.976356 %
4	23	0.589333	0.387317	0.474824	9.822212 %
5	41	0.602577	-0.247673	0.609302	10.042958 %
6	55	1.058527	-1.058527	0.456071	17.642121 %
7	23	0.702453	-0.679238	0.558782	11.707552 %
	162	0.909015	-0.264660	0.853203	15.150256 %

rem : all differences are calculated by doing TI-QPI

Filename : dage7test.stattix

Average QPI for the document : 5.580402
Average TI for the document : 5.165120
Average difference between TI and QPI : -0.415282
Total number of sentences taken into account : 398/414

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	3.317344	2.734762	1.676487	55.289063 %
3	23	1.878740	1.878740	0.435131	31.312337 %
4	53	0.950287	0.941324	0.335370	15.838111 %
5	73	0.378091	-0.045911	0.380955	6.301522 %
6	143	0.793066	-0.789703	0.359274	13.217771 %
7	101	1.572279	-1.542359	0.532634	26.204647 %
	398	1.030081	-0.415282	0.926042	17.168017 %

rem : all differences are calculated by doing TI-QPI

Filename : papert.stattix

Average QPI for the document : 4.410959
Average TI for the document : 5.327165
Average difference between TI and QPI : 0.916206
Total number of sentences taken into account : 73/78

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	6	2.163262	2.163262	0.823205	36.054369 %
3	15	2.071312	2.071312	0.466252	34.521874 %
4	13	1.486706	1.486706	0.341021	24.778427 %
5	23	0.567531	0.443156	0.346255	9.458850 %
6	14	0.368832	-0.336168	0.289851	6.147193 %
7	2	0.989819	-0.989819	0.092672	16.496975 %
	73	1.144835	0.916206	0.904899	19.080576 %

rem : all differences are calculated by doing TI-QPI

****

Data about the file : engnote1

****

Mean Observed QPI : 5.083601
Mean Calculated TI : 5.161653
Difference (TI-QPI) : 0.078052

	Factor Name	Total	Average	weights	Avg*weights
	Constant			1.919707	
FC 1	Number of the Sentence	53608	172.372990	0.000000	0.000000
FC 2	Qpi	0	0.000000	0.000000	0.000000
FC 3	Length	5593	17.983923	0.000000	0.000000
FC 4	Broken Sentences	0	0.000000	0.000000	0.000000
FC 5	Short Parens	14	0.045016	-0.311603	-0.014027
FC 6	Long Parens	9	0.028939	-0.078235	-0.002264
FC 7	Coord. Conj	105	0.337621	-0.040004	-0.013506
FC 8	Homographs	2289	7.360129	0.076142	0.560415
FC 9	No Verb	34	0.109325	-0.027959	-0.003057
FC 10	All Upper Case	11	0.035370	-0.765625	-0.027080
FC 11	Source language	622	2.000000	0.000000	0.000000
FC 12	Marked Interrog	34	0.109325	0.523485	0.057230
FC 13	WH Interrog	6	0.019293	-0.181528	-0.003502
FC 14	Suspicious EOS	0	0.000000	-1.248151	-0.000000
FC 15	SWORK to Verb	823	2.646302	0.001107	0.002929
FC 16	Dble-quoted Verb SWORK	4	0.012862	0.787702	0.010131
FC 17	Unmatched Left Parenth	2	0.006431	1.266236	0.008143
FC 18	Not WC20 Count	4732	15.215434	0.006576	0.100057
FC 19	All Initial Caps	10	0.032154	-0.415790	-0.013369
MX 1	Dependent Clause	89	0.286174	0.054209	0.015513
MX 2	DC Secondary	7	0.022508	0.204358	0.004600
MX 3	Nested Parens in REL	4	0.012862	-0.495596	-0.006374
MX 4	REL Clauses	114	0.366559	0.245406	0.089956
MX 5	Commas	160	0.514469	-0.026418	-0.013591
MX 6	Tagged Interrog	1	0.003215	1.422493	0.004574
MX 7	TH Clauses	56	0.180064	0.132379	0.023837
MX 8	Unfound Words	63	0.202572	0.341173	0.069112
SS 1	No Main Clause	7	0.022508	0.000000	0.000000
SS 2	Compound Clause	85	0.273312	0.047160	0.012889
SS 3	INGs in Sent.	31	0.099678	0.152401	0.015191
SS 4	ING Clause	55	0.176849	0.048572	0.008590
SS 5	AS	31	0.099678	0.096449	0.009614
SS 6	BOS AS	1	0.003215	0.468432	0.001506
SS 7	WITH	25	0.080386	0.192842	0.015502
SS 8	Long NP	62	0.199357	-0.013985	-0.002788
SS 9	ING NP	6	0.019293	0.238871	0.004608
SS 10	Quotes/Bold/Etc.	0	0.000000	0.000000	0.000000
SS 11	Initial CapStrings	1	0.003215	0.876528	0.002818
SS 12	All Upper Case_Strings	1	0.003215	0.305648	0.000983
SS 13	Does not exist	0	0.000000	0.000000	0.000000
SS 14	Does not exist	0	0.000000	0.000000	0.000000
SS 15	Does not exist	0	0.000000	0.000000	0.000000
SS 16	Does not exist	0	0.000000	0.000000	0.000000
				8 - TI =	2.838347

****

Data about the file : microsof

****

Mean Observed QPI : 5.535088
Mean Calculated TI : 5.225130
Difference (TI-QPI) : -0.309958

	Factor Name	Total	Average	weights	Avg*weights
	Constant			1.919707	
FC 1	Number of the Sentence	6753	59.236842	0.000000	0.000000
FC 2	Qpi	0	0.000000	0.000000	0.000000
FC 3	Length	2073	18.184211	0.000000	0.000000
FC 4	Broken Sentences	0	0.000000	0.000000	0.000000
FC 5	Short Parens	8	0.070175	-0.311603	-0.021867
FC 6	Long Parens	2	0.017544	-0.078235	-0.001373
FC 7	Coord. Conj	53	0.464912	-0.040004	-0.018598
FC 8	Homographs	827	7.254386	0.076142	0.552363
FC 9	No Verb	0	0.000000	-0.027959	-0.000000
FC 10	All Upper Case	0	0.000000	-0.765625	-0.000000
FC 11	Source language	228	2.000000	0.000000	0.000000
FC 12	Marked Interrog	0	0.000000	0.523485	0.000000
FC 13	WH Interrog	0	0.000000	-0.181528	-0.000000
FC 14	Suspicious EOS	0	0.000000	-1.248151	-0.000000
FC 15	SWORK to Verb	291	2.552632	0.001107	0.002826
FC 16	Dble-quoted Verb SWORK	1	0.008772	0.787702	0.006910
FC 17	Unmatched Left Parenth	0	0.000000	1.266236	0.000000
FC 18	Not WC20 Count	1710	15.000000	0.006576	0.098640
FC 19	All Initial Caps	4	0.035088	-0.415790	-0.014589
MX 1	Dependent Clause	57	0.500000	0.054209	0.027105
MX 2	DC Secondary	10	0.087719	0.204358	0.017926
MX 3	Nested Parens in REL	0	0.000000	-0.495596	-0.000000
MX 4	REL Clauses	38	0.333333	0.245406	0.081802
MX 5	Commas	105	0.921053	-0.026418	-0.024332
MX 6	Tagged Interrog	0	0.000000	1.422493	0.000000
MX 7	TH Clauses	8	0.070175	0.132379	0.009290
MX 8	Unfound Words	25	0.219298	0.341173	0.074819
SS 1	No Main Clause	10	0.087719	0.000000	0.000000
SS 2	Compound Clause	24	0.210526	0.047160	0.009928
SS 3	INGs in Sent.	19	0.166667	0.152401	0.025400
SS 4	ING Clause	33	0.289474	0.048572	0.014060
SS 5	AS	12	0.105263	0.096449	0.010153
SS 6	BOS AS	0	0.000000	0.468432	0.000000
SS 7	WITH	0	0.000000	0.192842	0.000000
SS 8	Long NP	30	0.263158	-0.013985	-0.003680
SS 9	ING NP	4	0.035088	0.238871	0.008381
SS 10	Quotes/Bold/Etc.	0	0.000000	0.000000	0.000000
SS 11	Initial CapStrings	0	0.000000	0.876528	0.000000
SS 12	All Upper Case_Strings	0	0.000000	0.305648	0.000000
SS 13	Does not exist	0	0.000000	0.000000	0.000000
SS 14	Does not exist	0	0.000000	0.000000	0.000000
SS 15	Does not exist	0	0.000000	0.000000	0.000000
SS 16	Does not exist	0	0.000000	0.000000	0.000000
				8 - TI =	2.774870

****
Data about the file : nordson
****

Mean Observed QPI : 4.906977
Mean Calculated TI : 5.283513
Difference (TI-QPI) : 0.376537

	Factor Name	Total	Average	weights	Avg*weights
	Constant			1.919707	
FC 1	Number of the Sentence	33411	129.500000	0.000000	0.000000
FC 2	Qpi	0	0.000000	0.000000	0.000000
FC 3	Length	4829	18.717054	0.000000	0.000000
FC 4	Broken Sentences	0	0.000000	0.000000	0.000000
FC 5	Short Parens	29	0.112403	-0.311603	-0.035025
FC 6	Long Parens	4	0.015504	-0.078235	-0.001213
FC 7	Coord. Conj	131	0.507752	-0.040004	-0.020312
FC 8	Homographs	1865	7.228682	0.076142	0.550406
FC 9	No Verb	40	0.155039	-0.027959	-0.004335
FC 10	All Upper Case	6	0.023256	-0.765625	-0.017805
FC 11	Source language	516	2.000000	0.000000	0.000000
FC 12	Marked Interrog	0	0.000000	0.523485	0.000000
FC 13	WH Interrog	0	0.000000	-0.181528	-0.000000
FC 14	Suspicious EOS	1	0.003876	-1.248151	-0.004838
FC 15	SWORK to Verb	1079	4.182171	0.001107	0.004630
FC 16	Dble-quoted Verb SWORK	1	0.003876	0.787702	0.003053
FC 17	Unmatched Left Parenth	0	0.000000	1.266236	0.000000
FC 18	Not WC20 Count	4032	15.627907	0.006576	0.102769
FC 19	All Initial Caps	27	0.104651	-0.415790	-0.043513
MX 1	Dependent Clause	78	0.302326	0.054209	0.016389
MX 2	DC Secondary	0	0.000000	0.204358	0.000000
MX 3	Nested Parens in REL	1	0.003876	-0.495596	-0.001921
MX 4	REL Clauses	72	0.279070	0.245406	0.068485
MX 5	Commas	177	0.686047	-0.026418	-0.018124
MX 6	Tagged Interrog	1	0.003876	1.422493	0.005514
MX 7	TH Clauses	9	0.034884	0.132379	0.004618
MX 8	Unfound Words	70	0.271318	0.341173	0.092566
SS 1	No Main Clause	0	0.000000	0.000000	0.000000
SS 2	Compound Clause	51	0.197674	0.047160	0.009322
SS 3	INGs in Sent.	44	0.170543	0.152401	0.025991
SS 4	ING Clause	31	0.120155	0.048572	0.005836
SS 5	AS	6	0.023256	0.096449	0.002243
SS 6	BOS AS	4	0.015504	0.468432	0.007263
SS 7	WITH	23	0.089147	0.192842	0.017191
SS 8	Long NP	168	0.651163	-0.013985	-0.009107
SS 9	ING NP	23	0.089147	0.238871	0.021295
SS 10	Quotes/Bold/Etc.	0	0.000000	0.000000	0.000000
SS 11	Initial CapStrings	0	0.000000	0.876528	0.000000
SS 12	All Upper Case_Strings	13	0.050388	0.305648	0.015401
SS 13	Does not exist	0	0.000000	0.000000	0.000000
SS 14	Does not exist	0	0.000000	0.000000	0.000000
SS 15	Does not exist	0	0.000000	0.000000	0.000000
SS 16	Does not exist	0	0.000000	0.000000	0.000000
				8 - TI =	2.716487

Filename : clin2test.stattix

Average QPI for the document : 5.222222
Average TI for the document : 4.820059
Average difference between TI and QPI : -0.402163
Total number of sentences taken into account : 162/176

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	2	1.577784	1.577784	0.032885	26.296392 %
3	18	1.613284	1.409440	0.735386	26.888065 %
4	23	0.507182	0.171163	0.467802	8.453032 %
5	41	0.695643	-0.403382	0.683334	11.594057 %
6	55	1.182309	-1.182309	0.426699	19.705148 %
7	23	0.973260	-0.697696	0.854682	16.221002 %
	162	0.986378	-0.402163	0.894550	16.439631 %

rem : all differences are calculated by doing TI-QPI

Filename : dage7test.stattix

Average QPI for the document : 5.583960
Average TI for the document : 4.979679
Average difference between TI and QPI : -0.604281
Total number of sentences taken into account : 399/414

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	5	2.974627	2.530764	1.456169	49.577123 %
3	23	1.525339	1.497133	0.638432	25.422311 %
4	53	0.695707	0.573613	0.513976	11.595118 %
5	73	0.446870	-0.227916	0.441412	7.447827 %
6	143	1.059216	-0.990172	0.497447	17.653607 %
7	102	1.647635	-1.572206	0.556240	27.460591 %
	399	1.100192	-0.604281	0.925624	18.336535 %

rem : all differences are calculated by doing TI-QPI

Filename : papert.stattix

Average QPI for the document : 4.445946
Average TI for the document : 4.937839
Average difference between TI and QPI : 0.491893
Total number of sentences taken into account : 74/78

QPI	#sent.	Mean Abs. Diff.	Mean difference	Mean Abs. Dev.	Error Perc.
1	0	NaN	NaN	NaN	NaN %
2	6	1.694322	1.694322	0.740014	28.238697 %
3	15	1.664394	1.664394	0.559812	27.739892 %
4	13	0.991446	0.984836	0.409100	16.524097 %
5	23	0.401189	0.069119	0.398956	6.686480 %
6	14	0.665915	-0.665915	0.364020	11.098585 %
7	3	1.267188	-1.267188	0.114897	21.119806 %
	74	0.950978	0.491893	0.907726	15.849628 %

rem : all differences are calculated by doing TI-QPI